Modelling Temporal Patterns in User Behaviour

Von der Fakultät für Ingenieurwissenschaften,
Abteilung Informatik und Angewandte Kognitionswissenschaft
der Universität Duisburg-Essen

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

genehmigte Dissertation

von

Dipl.-Inform. Sebastian Dungs
aus
Essen

1. Gutachter: Prof. Dr.-Ing. Norbert Fuhr
2. Gutachter: Prof. Dr.-Ing. Andreas Nürnberger

Datum der mündlichen Prüfung: 05. März 2019

Abstract

Modelling sequential data is one of the most challenging problems in machine learning research. The object of the investigation can be records of user behaviour, which are analysed to uncover hidden temporal patterns. A broad range of solutions exist for this type of problem, including multi-space hidden Markov models (HMMs). The main strength of this technique is its ability to jointly model features on a discrete and continuous scale, which is a property that conventional HMMs do not possess; therefore, multi-space HMMs are well suited to model temporal patterns in combination with other features. However, so far, they have not been utilised to build temporal models of user behaviour. Based on a newly developed integrated framework for creating multi-space HMMs, user behaviour is modelled in two fields of research. By creating HMMs of two phases in user behaviour during a session search, prior qualitative information-seeking models are augmented by a quantitative component. In a series of experiments based on a search engine transaction log, it could be shown that approximately one out of three search sessions reached the second phase, which is characterised by heightened effectiveness and efficiency of user actions. Furthermore, how the search phase model can be used to estimate crucial parameters of a search session is demonstrated; for example, the expected time to find the next relevant document. In the second practical application, the HMM framework's versatility is highlighted by utilising the models as a classifier to detect rumourous conversations on Twitter and to model their veracity. Thus, this work complements prior research by using tweet stance and time as the only features to build a high recall rumour detection system based on multi-space HMMs. Especially when modelling rumour veracity, the strength of the joint modelling of the temporal component is evident since the multi-space HMMs achieve state-of-the-art results. In further experiments, it is also shown that the models are robust to noise and can provide timely veracity classifications.

Kurzzusammenfassung

Die Modellierung von sequentiellen Daten ist eine der großen Herausforderungen bei der Erforschung von maschinellen Lernverfahren. Gegenstand der Untersuchung können dabei auch Aufzeichnungen von Benutzerverhalten sein, welche analysiert werden, um versteckte zeitliche Muster zu entdecken. Eine Vielzahl möglicher Lösungen für derartige Probleme ist bekannt, eine davon sind mehrräumige Hidden-Markov-Modelle (HMM). Der Vorteil dieser Technologie ist ihre Fähigkeit, diskrete und kontinuierliche Merkmale in einem vereinten Modell abzubilden. Dies ist eine Eigenschaft, welche herkömmliche HMM nicht besitzen. Aus diesem Grund sind mehrräumige HMM besonders gut zur Modellierung von zeitlichen Mustern in Kombination mit anderen Merkmalen geeignet. Bisher wurden sie allerdings noch nicht dazu genutzt, um zeitliche Modelle von Benutzerverhalten zu erstellen. Basierend auf einem neu entwickelten Framework zur Erstellung von mehrräumigen HMM, wird hier das Benutzerverhalten in zwei Anwendungsgebieten modelliert: Durch die Erstellung von HMM, welche zwei Phasen einer Suchsitzung beschreiben, werden vorhergehende Information-Seeking-Modelle um eine quantitative Komponente ergänzt. In einer Experimentalreihe, basierend auf einem Suchmaschinen-Transaktionslog, wird anschließend gezeigt, dass in circa einer von drei Suchsitzungen die zweite Phase erreicht wird, welche von erhöhter Effektivität und Effizienz geprägt ist. Außerdem wird erörtert, wie das Zweiphasenmodell dazu benutzt werden kann, um kritische Parameter der Suchsitzung zu bestimmen, wie etwa die voraussichtlich benötigte Zeit bis zum Finden des nächsten relevanten Dokuments. Die zweite praktische Anwendung des HMM Frameworks demonstriert dessen Flexibilität, indem die Modelle benutzt werden, um Gerüchte auf Twitter zu erkennen und deren Wahrhaftigkeit zu bestimmen. In diesem Fall ergänzt diese Arbeit die vorhergehenden zu dem Thema, indem nur der Standpunkt eines Tweets sowie dessen Zeitpunkt als Merkmal benutzt wird, um mit Hilfe von mehrräumigen HMM Gerüchte mit hoher Trefferquote zu erkennen. Besonders beim Modellieren der Wahrhaftigkeit zeigt sich die Stärke des vereinten Modells mit zeitlicher Komponente, da die mehrräumigen HMM Ergebnisse auf dem Niveau des neusten Stands der Technik erzielen. In weiteren Experimenten wird außerdem gezeigt, dass diese Modelle robust gegenüber Störungen in den Ausgangsdaten sind und bereits auf Basis von sehr kurzen Beobachtungssequenzen brauchbare Ergebnisse liefern.

Contents

j

CHAPTER 1

Introduction

Machine learning is one of the most rapidly progressing research areas in the literature. The automated extraction of knowledge from vast amounts of data has not only penetrated into fields other than computer science, but it has also changed processes in stable industrial sectors, such as the automotive industry. In general, the term machine learning is used in a rather broad context and can refer to many different approaches specific to individual application domains. In practice, the most appropriate learning algorithms are often predetermined by the type of data at hand and the nature of the desired output.

One of the common data categories is sequential or time-series data, in which measurements are repeatedly taken over time and stored as a sequence. What sets this kind of data apart from other types is the fact that the full informational value contained in the data only becomes available when the raw measurements are considered jointly with the time of measurement. All algorithmic solutions for modelling sequential data must ensure that the order of observations is maintained through the model training process as well as when making predictions based on the data, which makes sequential data one of the most challenging problems in machine learning research (Yang and Wu, 2006).

Sequential data have arisen in a large number of domains, ranging from stock value progression in economics to measurements of atmospheric pressure distributions in meteorology. Other domains include astronomy, where starlight fluctuations are measured to spot exoplanets, or engineering, where levels in power grid

workloads are monitored. Naturally, sequential data are also prevalent in computer science applications, for example, while considering network intrusion prevention systems or managing network routing. Furthermore, automated speech recognition and synthesis is a research and application domain that exclusively deals with sequential data.

Although sequential data are prevalent in all the aforementioned examples, the actual learning problems associated with these data can be of different principle categories. Given the nature of the domain, the overall goal can be to determine the most likely next element in a sequence (for example in stock value prediction) or to perform sequence classification (for example when analysing network activity patterns for anomalies). The goal of the application can also be the generation of new sequential data following the structure of previously recorded data, as is for example common in real-time text suggestion systems that are used for typing on mobile devices. As an extension of the last category, sequence-to-sequence prediction has the goal of generating entirely new sequences which do not necessarily need to follow the structure of the previously observed sequences, such as the summarisation of large texts. Despite the concrete outputs produced by these methods, in any case the raw measurements are first abstracted into sequential models. Afterwards, inference can be drawn about the present state of the system, which is then used to make predictions about its future development.

There exists a large number of concrete algorithmic implementations to create models from sequential data. In recent years, deep learning approaches have been used widely, which work particularly well when the amount of available training data is very large or when knowledge about the data is limited (Längkvist et al., 2014; Lipton et al., 2015). However, there is a number of 'traditional' machine learning algorithms that can also be applied to sequential data. In his survey paper Dietterich (2002) explicitly mentions (recurrent) sliding-window methods, maximum entropy models, conditional random fields, graph transformer networks and hidden Markov models (HMMs), all of which have their respective strengths depending on the application.

As the large number of examples given above demonstrate, sequential data can originate from various sources and may feature different properties that need to be regarded when creating models. In this thesis, the focus is going to be on sequences generated by humans in interactions with computer systems—either implicitly (for example data derived from interaction logs) or through explicit statements—in order to model the associated user behaviour patterns. The key difference between this task and others based on different kinds of sequential data is the fact that the users' behaviour is based on cognitive processes which are invariably *hidden* to the observer. Therefore, modelling user behaviour requires learning mechanisms that can deal with uncertainty and provide means to extract and model hidden information contained implicitly in the observable sequences.

Hidden Markov models are well suited for the task of user-centred sequence modelling. As the name suggests, one major aspect of these models is that they include a hidden process, whose parameters are learned based on the sequential observations. Consequently, HMMs can be used to describe the cognitive processes that users undergo when interacting with a system, for example, based on their clicks or performed actions. Since HMMs are probabilistic, incomplete knowledge about a hidden process does not obstruct model generation, whereas other approaches require the practitioner to completely specify the system to be modelled.

Hidden Markov models also have another advantage: they are comparably simple in their architecture, allowing explicit specification of a limited set of features on which the modelling should be based. In contrast to this are the deep learning approaches where a large number of features are created during model generation, which are hard to interpret analytically. In the latter approaches, it is likely that any predictions being made by deep models are not based on the same features as those on which humans would base their judgements. Although deep models have begun to surpass HMMs in some prominent application domains, such as speech synthesis, in terms of achieved results, HMMs are still useful when a thorough understanding of a hidden process is desired. Lastly, HMMs are well suited for problems where the size of the visible input vocabulary is limited and remains fixed throughout all observed sequences. Yet, simultaneously, the hidden process can be of almost arbitrary complexity. This was for example impressively demonstrated in the human genome project, where the entire human gene structure was decoded using HMMs trained on sequences of the four main nucleotides (Venter et al., 2001).

In the following sections this thesis' main research goal and contribution is introduced. Subsequently, two practical applications of the developed methods are motivated, which are addressing relevant research gaps in their respective domains.

1.1 Contribution

Within the frame of this thesis, a flexible framework for modelling sequential data is presented, which in particular is well suited for detection and description of human behavioural patterns. By utilising hidden Markov models the models' hidden states can be used to represent the unknown cognitive conditions and modelling their properties allows to understand the reasoning processes humans undergo when interacting with information systems.

There are a number of further reasons why HMMs were chosen as the basis of

the modelling framework: First, out of the afore-mentioned modelling techniques inherently well suited for modelling sequential data, HMMs provide a reasonable trade off between models' complexity and expressiveness. When considering possible limitations in availability of user behavioural data, hidden Markov models have the advantage that model training can also be performed based on limited data samples when models are constrained in terms of the number of involved parameters. Additionally, their training requires comparably low amounts of computational resources, especially in comparison to the prevalent deep models.

However, it is important to note that HMMs have a transparent structure and all their parameters are known anytime which allows them to be purposefully manipulated if the specific situation requires. Therefore, this type of models is well suited when prior knowledge about the specific application domain is available. In this sense, the developed framework is also composed as an expert tool, which allows practitioners to address specific research questions, where explicit integration of domain knowledge is desired and features have a natural interpretation.

One of the frameworks' key technical aspects is the ability to explicitly include observations' times into the modelling process, extending on those approaches where only the chronological order of the sequences' elements is considered. By doing so, sequential behavioural data can be explored in depth for underlying hidden temporal patterns. Prior to this work, there existed no general purpose implementation for this task based on hidden Markov models. More particularly, implementations generally focus on the simplest version of hidden Markov models, which are limited to model a fixed number of discrete categorical observations.

More elaborate implementations of *continuous* hidden Markov models allow the specification of a continuous range of observation values and are able to learn the relationship between observations and hidden states based on probability density functions. In principle, such a continuous scale could also be used to model progression of time. However, even continuous HMMs are too restrictive to adequately model user behaviour when other features need to be considered simultaneously. Specifically, there exists no HMM framework capable of describing discrete and continuous features in a joint model. Although a number of methods exist to transform discrete to continuous data or vice versa, any of these transformations potentially results in loss of information which makes these solutions undesirable in the general case (Witten et al., 2016, Chapter 8).

Therefore, the framework developed in this work in particular also includes an implementation of the mathematical concepts of *multi-space* hidden Markov models. This extension of HMMs was originally introduced by Tokuda et al. (1999) in the context of automated natural language speech recognition and synthesis. Naturally, researchers in this domain have high proficiency in dealing with temporal data and also found the need to simultaneously model features on discrete and

continuous scale. In a multi-space model all features are described in n-dimensional real space. The models' increased expressiveness comes from the fact that dimensionality of every feature can be specified independently. Furthermore, any feature can also have zero-dimensionality, which collapses the feature's space to a discrete scale. Therefore, a multi-space HMM is not only an extension of discrete and continuous models, but notably it can be used to model features on both scales simultaneously without the need for data transformations.

Despite their superior modelling power, multi-space HMMs have seldom been applied in practice and none of the prior works explored their usage for modelling temporal patterns in user behaviour.

In this work, the newly developed HMM framework is used to implement models of user behaviour on the basis of discrete, continuous and multi-space models. The practical application of the framework's different modelling types allows to investigate the question how the addition of the temporal component as a further explicit feature can deepen the understanding about sequences of behavioural data compared to other approaches limited to considering their conventional feature(s).

In the next sections, the concrete application examples are introduced while elaborating on the research gaps that are addressed. In these applications the principal fact that HMMs are *generative* models is exploited to perform sequence prediction, while also considering the models' prospective use for sequence generation. Furthermore, the models are applied to perform sequence classification tasks, a principal application field where prior HMM usage is yet scarce.

1.2 Applications and Motivation

Two fields for application of the framework were explored in the scope of this thesis, both of which focussing on the modelling of user behavioural data. However, the applications differ not only in data source and dataset size, but also in terms of how and to what end the data is utilised. In the first application discrete and continuous HMMs are build based on a social science search engine's transaction log including more than one million entries. In this case the overall goal is to build a novel quantitative understanding of the hidden mental stages users undergo in the course of complex search sessions. It is worth pointing out that this application is an instance of *unsupervised* learning problems since it is based on real transaction logs, for which no ground truth information about the users' actual intents or tasks is available.

In the second application user behavioural data is captured differently in terms of collective stance information expressed in Twitter conversations. In a number of

supervised classification tasks discrete and multi-space models are used to identify hidden temporal patterns. Based on these patterns rumourous conversations are identified on Twitter while also successfully modelling the rumours' veracity. Abstracting from textual features and using collective stance information for these tasks is in contrast to the prior work in this field and has not been explored in combination with HMMs before.

Both applications of the framework were chosen in particular because they are addressing evident gaps in their respective research domain which are clarified below. The experimental results provide valuable contribution to the respective research communities, also acknowledged by the fact that they were published in well-known peer reviewed venues (Dungs et al., 2018; Dungs and Fuhr, 2017).

1.2.1 Search Phase Detection

The first application of the framework focusses on modelling user behaviour in interactive information retrieval (IIR). The classic system-oriented view of information retrieval, which is the content-oriented search in unstructured documents, assumes a static information need. However, IIR has a broader view on the problem and focusses on the users' interaction with the information systems while assuming a dynamic information need. In his early work, Ingwersen (1992) for example identifies improvements in information retrieval effectiveness as one of the main goals of IIR research. Furthermore, he states that empirical evaluations of users' searching behaviour can unveil common patterns whose discovery will lead to the development of search models that are beneficial to reaching the main research goal.

A number of models following the interactive information retrieval paradigm were created, some of which are dating from the analogue ages even preceding the work by Ingwersen. Wilson (1999) gives an overview of prominent early models of users' behaviour in complex search tasks, which are sometimes of a pure theoretical nature. Other models are limited to small scale empirical analysis based on data generated through direct observation of users' interactions when performing search tasks, sometimes complemented by structured interviews and questionnaires. Consequently, the body of IIR models are of varying detail, ranging from high level constructs to more situation-specific representations of the problem. However, although they are addressing similar issues in alternate ways, they are not necessarily in conflict to each other, as it is also emphasised by Wilson.

Well-known examples for empirical analysis of users behaviour are the works by Kuhlthau (1991) and Ellis (1989), which are also the most relevant studies in the scope of this thesis. Both researchers derived models which describe users' behaviour in complex tasks and claim existence of *search phases*, defined as proto-

typical states which describe a composition of users' mental as well as the search engine's state combined with the current work task. Furthermore, according to these models, users undergo an unknown sequence of state changes in the process of engaging in IIR, where phase transitions are stimulated by the search engine and user's mental work.

While these models are undoubtedly valuable as a foundation to the theoretical understanding of IIR user-system interaction, they are also of qualitative nature and build based on small-scale empirical studies. There is a need to supplement the early qualitative findings with quantitative data, ideally capturing real users' behaviour conducting non-artificial searches (Ageev et al., 2011). More recent attempts to include search phases into user models have been undertaken for example by Pharo and Nordlie (2012), who built a two-stage model of users' contextual topic knowledge in session search. Additionally, Huurdeman and Kamps (2014) investigated how modern search system interfaces could be modified to support multiple search phases.

Nevertheless, as of yet, quantitative models are still underused in IIR research. Few of the notable exceptions are for example the works by Han et al. (2013), who relate a hidden Markov model of search tactics to Marchionini's (1997) information seeking process model, or Yue et al.'s (2014) hidden Markov model of collaborative web search. Kotzyba et al. (2017) also use hidden Markov models to predict the task type of users' current search. However, although there is work acknowledging the existence of search phases as well as a number of quantitative user models, none of the prior works unifies the two concepts and builds a quantitative model of search phase progression.

Based on the HMM framework developed within this work, this research gap is addressed. This successful quantitative modelling of search phases is not only complementing prior work, but also providing notable potential for new applications. For example, one long-term goal could be the automatic classification of users' actions in search phases in real time during the search process. Not only would this be a substantial step towards estimating the parameters of the iPRP as proposed by Fuhr (2008). It would also advance research towards building search systems that are able to offer user guidance targeted at the specific situation a user is currently in, ultimately further increasing retrieval effectiveness as proposed already by Ingwersen (1992).

The contribution of this work is to provide a first step towards a quantitative search phase model based on real transaction logs in an unmoderated environment. While there exist prior research works on quantitative models, none of these have explored this method in context of the qualitative search phase models, while also including hidden states to model the unknown behavioural patterns. The general feasibility of such a model is demonstrated here.

1.2.2 Rumour Veracity Classification and Detection

In general potential input to the modelling framework is not limited to specific domains or data sources as long as the data can be expressed as a temporal ordered sequence of user actions which are quantifiable by a known and finite alphabet. Furthermore, generated models can on the one hand be the direct object of investigation, as in the search phase application described above, but on the other hand can also be used as intermediary tools to achieve subsequent tasks, for example classification.

To highlight this flexibility, the framework is also applied to the problems of *rumour detection* and *rumour veracity classification* on Twitter, where variants of HMMs are used to uncover underlying hidden temporal patterns in collective crowd stance.

Ensuring proper and timely response to potentially fast and wide spreading rumours is subject of many recent scientific studies—see Rubin (2017) for a broader review on the topic. Debunking rumours and fake news is also the goal of a number of dedicated investigative and journalistic websites and projects, for example Snopes[1], FactCheck[2] or FullFact[3]. Moreover, the topic's popularity is being highlighted by recent political developments and reinforced by mass media coverage in recent years, recognising rumours and fake news as one of the major challenges for citizens, journalists and organisations in today's media landscape in general.

Research on social media rumours usually focuses on data gathered from Twitter as it is comparably easy to acquire. Due to Twitter's unmoderated nature and its status as a primary address for real time news, there are ample events provoking users to post rumourous tweets. The high availability of Twitter data has also made it a prime subject for investigating a number of research questions, particularly from the field of natural language processing, among which one of the highest popularity is tweet stance classification (Procter et al., 2013). For example, the SemEval 2016 challenge on this topic single-handedly attracted 28 individual research contributions (Mohammad et al., 2016). In general, a tweet's stance can be regarded as the writer's opinion towards another tweet—which may potentially be rumourous—and can take the values *supporting*, *denying*, *questioning* and *commenting*.

Previously, Mendoza et al. (2010) could show in a study that true rumours were affirmed in more then 90% of the cases by the crowd while still 50% of false rumours were denied or questioned by other users. Apparently, users behave differently depending on the rumours' veracity or generally when being confronted

[1] `https://www.snopes.com`—last accessed 24.10.2018
[2] `https://www.factcheck.org`—last accessed 24.10.2018
[3] `https://fullfact.org`—last accessed 24.10.2018

with rumourous or non-rumourous content. Consequently, it is reasonable to assume that the opinions towards a statement expressed by others—captured in the form of tweet stance—can also be used as a feature to model rumours on Twitter. Specifically, one of the main assumptions underlying this second application is that by considering ordered sequences of tweets' stances as input to the framework, joint crowd wisdom can be exploited to classify rumours' veracity.

This approach is contrasting other automated veracity classification methods in the literature, as these commonly use text- and user-specific as well as propagation-based features for the task (Castillo et al., 2011; Ma et al., 2015; Vosoughi, 2015). User behavioural patterns are not explicitly considered in prior work, while stance expressed in tweets is only seldom used and if at all only conjointly with other features, as for example by Liu et al. (2015a).

However, to tackle the problem of rumour veracity classification, rumours need to be readily visible in the huge mass of social media postings. Apart from a few hand crafted datasets developed for building aforementioned classifiers, this is hardly ever the case in practice. Therefore, actual rumour detection is in fact a prerequisite in a larger pipeline when dealing with rumours in social media, as also Zubiaga et al. (2018) have pointed out previously.

In contrast to veracity classification, especially the detection of emerging rumours is a harder task that until recently has not attracted as much attention in the past, most notable exceptions being the work by Zhao et al. (2015) and Zubiaga et al. (2017). While the former approach uses hand-crafted regular expressions as keywords to flag potentially rumourous conversations, the latter uses a feature rich conditional random fields for the task. Despite the irrefutable promising results especially of the conditional random field approach, both methods are still improvable in terms of recall of rumourous conversations.

In some of the prior work on HMM-based classifications system these are shown to exhibit superior recall, also specifically when being compared to conditional random field approaches (Ponomareva et al., 2007). However, to the present day, HMMs have not been applied to the rumour detection task in the literature. Therefore, while once more following the assumption of evolving temporal patterns in stance distributions, the created HMM framework is also used to build a rumour detector, which is performing classification based on user behaviour abstracted into the collective stance feature. The high recall output of the modelling process can afterwards be supplied to additional methods to further investigate the conversations' veracity. Since both classifiers make direct use of the joint modelling of temporal patterns in user behavioural data, the results continue to emphasise the benefit of the methods proposed in this work.

1.3 Summary

In summary, this thesis covers the detection and modelling of temporal sequential patterns, one of the many sub-problems in machine learning. The particular, the focus is on models that describe users' behaviour and its progression over time using variants of hidden Markov models. From a technical point of view, this thesis provides a multi-purpose implementation of multi-space HMMs, which are capable of unifying discrete and continuous features in a joint model, in particular to also capture temporal patterns.

The frameworks' versatility is demonstrated in two practical applications of user behaviour modelling, using a search engine transaction log as well as tweet level stance annotations extracted from rumours conversations on Twitter. The thesis' contribution towards the research community is ensured not only because both applications address evident research gaps, but also because the provided framework can in the future be a valuable foundation for creation of HMM-based user behavioural models in a wide range of domains.

1.4 Outline

The remainder of this thesis is structured in two main parts. The initial part begins with a review of sequence modelling approaches based on the Markov property as well as related concepts, before comparing these approaches with respect to their suitability for modelling temporal patterns in user behaviour. After motivating the choice of technology, the concrete implementation of a general purpose framework for creating multi-space HMMs is introduced in the following chapter. Subsequently, in the second part of this thesis the framework is applied in two use cases. First, user behaviour in session search is modelled as a two-phase process. Additionally, in the second application, multi-space models of users' responses on Twitter are used to detected rumours and to determine rumour veracity.

Part I

Technical Foundations

CHAPTER 2

Sequence Modelling Approaches

In the this chapter some of the well established sequence modelling techniques are introduced along with giving practical application examples. First of all, the family of Markov models is discussed in detail following the topology shown in Table 2.1. Models of this family can be divided in autonomous and controlled variants describing a fully or partially observable system. Based on these properties the systems grow in complexity and expressiveness. While the autonomous and fully observable system is the simplest variant, the other end of the spectrum is covered by controlled systems which are also able to model hidden components.

Table 2.1: Markov model typology overview

System type	System state	
	Fully observable	Partially observable
Autonomous	Markov chain	Hidden Markov model
Controlled	Markov decision process	Partially observable Markov decision process

Additional to the Markov model family, related and alternative techniques for sequence modelling are presented briefly. The chapter concludes with a side-by-side comparison of the techniques' strengths and weaknesses.

2.1 Markov Chains

With his pioneering work on stochastic processes Andrey Andreyevich Markov (1906) laid the foundation for what later became known as Markov chains or is sometimes simply referred to as Markov models. Markov chains are conceptionally based on the stochastic processes' *memoryless* property which states that processes' future states' conditional probability distribution only depends on the current state of the underlying process—not on the states preceding it. This memoryless property is also called the *Markov property* and processes fulfilling it are said to be *markovian*. Furthermore, the memoryless property is sometimes referred to as the *Markov assumption*, especially in the context of Hidden Markov Models where—depending on the subject matter—it may be unknown or unverifiable whether the property actually applies.

One of the many practical examples of Markov chains is the PageRank algorithm (Brin and Page, 1998), where the importance of a web page is determined by the stationary vector of the Markov chain describing the hyperlinks between these pages. Figure 2.1 illustrates how the PageRank (PR) of a website can be determined using a Markov chain with equally distributed transition probabilities. Similarly, other random walk problems can also be expressed as a Markov chain.

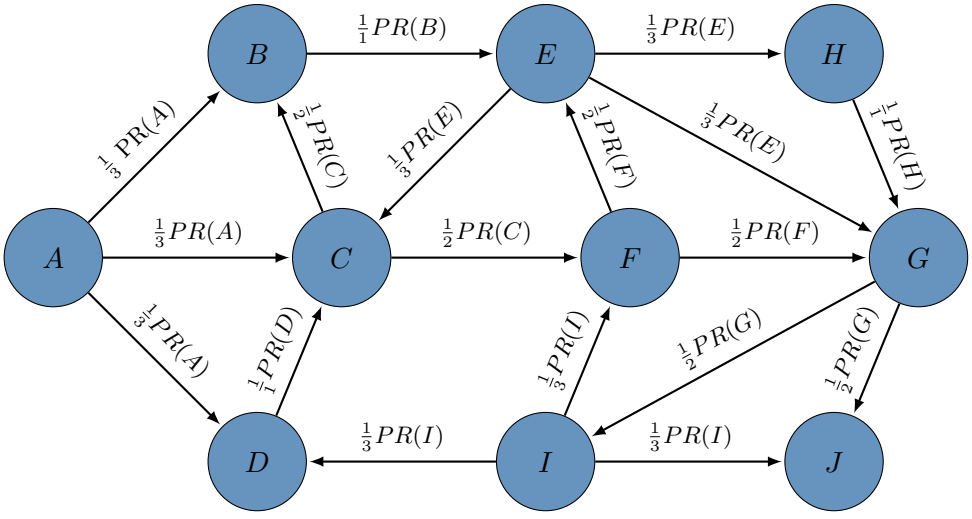


Figure 2.1: Simplified illustration of the PageRank algorithm as a Markov chain

Markov chains—and all related concepts discussed later in this chapter—can have varying formal definitions. Specifically, Markov chains can be defined in discrete or continuous time space with a countable state space or as having discrete time and continuous state space (Asmussen, 2003, Page 7). For example, Liu et al. (2008) use continuous time Markov processes, effectively extending the original PageRank algorithm to model users' browsing behaviour in web search more closely. However, for the scope of this thesis time and state space are always assumed to be discrete, following the formal definition given below:

A discrete Markov chain describes an observable random process $\{X_t\}_{t\in\mathbb{N}}$ possessing the Markov property, meaning transition probabilities satisfy the condition

$$\mathbb{P}(X_{t+1} = x | X_1 = x_1, X_2 = x_2, \ldots, X_t = x_t) = \mathbb{P}(X_{t+1} = x | X_t = x_t) \tag{2.1}$$

given that $\mathbb{P}(X_1 = x_1, X_2 = x_2, \ldots, X_t = x_t) > 0$. Furthermore, it is assumed that the set of possible states $x_i \in X$ composing the state space S is countable. Then, the transition probability distribution can be expressed as a stochastic matrix where the $(i, j)th$ element is given by

$$p_{ij} = \mathbb{P}(X_{t+1} = j | X_t = i). \tag{2.2}$$

Given the method's age, naturally, there exists an enormous body of research based on Markov chains. There has also been prior work on using Markov models to describe user behaviour in the information retrieval domain. For example, in their work on modelling search times Tran and Fuhr (2012, 2013) show how Markov models could in principle be used to estimate the parameters of the interactive probability ranking principle (Fuhr, 2008). Later Tran et al. (2017) extend on that method by introducing personalisation, significantly improving time estimates. However, so far their work is restricted to small scale lab studies.

Hassan et al. (2010) on the other hand build Markov models based on transaction logs from the Yahoo! search engine to predict search success. After training the model on the log data, the authors were able to identify a number of interaction patterns that are likely to be indicative of search success. Hassan et al.'s (2010) findings were later also confirmed by Ageev et al. (2011) using a different dataset and conditional random fields (see Section 2.6).

The strictness of the Markov property can also be relaxed by introducing higher-order chains. To be exact, a n^{th}-order Markov chain is a process with memory of length n, i.e. the future state depends on the past n states. Practical examples are the work by Xie and Joo (2010), who use fifth-order Markov Chains in a small scale lab study to model a number of general web search tactics. Similarly, but on a much larger scale, Chen and Cooper (2002) use semi-Markov chains to model

users' movement from one task to another. Like Xie and Joo the authors find higher-order Markov chains to best fit the data.

Generally, it is possible to construct higher-order Markov chains from the first-order Markov chains discussed above. However, doing so introduces practical difficulties as the required number of parameters increases exponentially with order n. Additionally to the associated computational cost, a higher parameter number also makes their estimation much more challenging. Especially if the training data is limited, state transition probabilities have to be determined based on few examples, potentially leading to overfitting and thus impairing the reliability of the outcome. Consequently, in the following higher-order Markov models are not further regarded.

2.2 Hidden Markov Models

To apply simple Markov models, i.e. Markov chains, the subject to be modelled is required to be completely known (visible). However, in many practical scenarios this requirement is to restrictive and instead hidden Markov models are used. HMMs extend Markov chains by introducing hidden states—representing those system properties that are not directly visible. While the model changes state (X_n) it produces visible output (Y_n) which is referred to as *emissions* or *observation* (see Figure 2.2). Since each state has its own probability distribution over the possible emissions, a sequence of emissions generated by an HMM contains information about its hidden state changes.

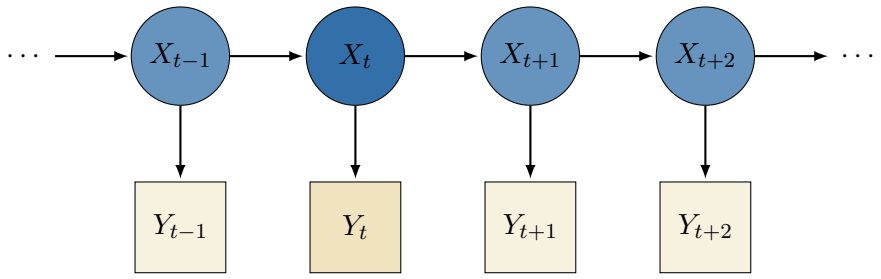


Figure 2.2: Trellis diagram of the general HMM architecture at time t

Looking at the historical development of HMMs, it was Stratonovich (1960) who first described some of the fundamental HMM-related mathematical procedures. In a series of papers by Baum and Petrie (1966), Baum and Eagon (1967) and Baum et al. (1970) these concepts were subsequently further developed. After a first surge of speech recognition systems based on simple discrete emissions (Baker, 1975; Jelinek et al., 1975), it was mainly the tutorial paper by Rabiner

(1989) which ultimately led to wide spread application also of more complex and powerful HMMs based on continuous emissions—e.g. described by Gaussian distributions—in a wide set of domains not limited to speech recognition.

Similar to Markov chains, HMMs can be defined in various forms of different complexity and expressiveness. In the following discrete time step random processes are assumed, which is also the most common definition. However, alternatively there are also variants of continuous time HMMs, where small time intervals also make for small probabilities of changing the state and state changes may occur without triggering any observable emission. An example of successful application of continuous time HMMs is disease progression detection as described by Liu et al. (2015b). However, learning parameters of continuous time HMMs introduces considerable difficulties compared to discrete time step HMMs and therefore this type of models is not further explored in the following.

In the following sections HMMs are introduced formally, first assuming observed signals are single categorical values belonging to a finite alphabet of values. Next, the definition is extended to continuous emissions and finally, several principal inference problems associated with HMMs are discussed.

Formally a HMM describes a system of two discrete random processes

$$\{X_t\}_{t\in\mathbb{N}} \text{ and } \{Y_t\}_{t\in\mathbb{N}} \tag{2.3}$$

of which only the latter is directly observable while the former is hidden. A HMM is defined as a quintuple

$$\theta = (S, E, A, B, \pi) \tag{2.4}$$

where $S = \{s_1, \ldots, s_n\}$ describes the set of N hidden states and $E = \{e_1, \ldots, e_m\}$ is the set of M possible categorical observations (the system's emissions). The hidden state transition matrix $A \in \mathbb{R}^{n\times n}$ is defined so that

$$a_{ij} = \mathbb{P}(X_t = s_j | X_{t-1} = s_i) \text{ with } a_{ij} \geq 0 \text{ and } \sum_i a_{ij} = 1. \tag{2.5}$$

The emission probability matrix $B \in \mathbb{R}^{n\times m}$ is defined so that

$$b_i(e_j) = \mathbb{P}(Y_t = e_j | X_t = s_i) \text{ with } b_i(e_j) \geq 0 \text{ and } \sum_j b_i(e_j) = 1. \tag{2.6}$$

Finally, $\pi \in \mathbb{R}^n$ is the start vector where $p_i = \mathbb{P}(X_1 = s_i)$ defines the states' probability of being the starting state at t_0.

A practical example illustrating this configuration is the dice tossing game, where an observer has to make a guess about a die secretly chosen by the tosser. Here, the set of n potentially unfair dice corresponds to the hidden states X and the observable emissions E is the set of the different number of pips on the dice's sides. The emission probability matrix B could then simply be set uniformly to $b(e_i) = \frac{1}{6}$ for all $e_i \in E, 1 \leq i \leq 6$ when using fair standard hexahedron die. When unfair dice are used, any other arbitrary distribution of pip probability can be chosen representing the dice's particularities. While the actual sequence of chosen die $X = (x_1, x_2, \ldots, x_n), x_i \in S$ is unknown to an observer, he or she can make inference about the tosser's choice given the sequence $O = (o_1, o_2, \ldots, o_n), o_i \in E$ of observed pips on the die and the emission probability matrix B.

Discrete emission HMMs are well suited for simple applications where the scope is restricted. The limits of this method are for example explored by the early work of Lane (1999), who used in total 2500 discrete symbols to built a model of users' legitimate interaction patterns while using a command line interface. Afterwards, this model is used to classify newly observed interactions as potentially malicious or rightful. However, in many practical scenarios features can be even more complex, for example, by being drawn from an infinite continuous value range. While in theory it may be possible to code continuous signals to a set of categorical values, such a transformation is inevitably associated with a loss of information. Furthermore, using such codebook approach quickly generates huge emissions sets which make calculations computationally demanding and models difficult to interpret.

2.2.1 From Discrete to Continuous Emissions

To overcome afore mentioned problems, a continuous hidden Markov model can be defined so that observation alphabet E consists of an uncountable infinite number of possible values o. In a continuous model the states' emissions are determined by continuous probability density functions (PDFs). In general, a continuous PDF is an integrable function $f(x)$, where $f(x) > 0$ and $\int f(x)dx = 1$. When such a PDF is used as an emission function in a HMM, the area under the curve is spanning the entire emission range E and it is deemed that all $x \in E$. To ensure the model's parameters can be estimated consistently, the continuous emission probability matrix is written as $B = \{b_i(\cdot)\}_{i=1}^{N}$ using the PDFs most appropriate for the data. It is common practice to use a set of finite mixture continuous PDFs to describe the probability of observing any value o. In this

case elements of emission probability matrix B take the form

$$b_i(\boldsymbol{o}) = \sum_{m=1}^{M} c_{im} \mathfrak{N}[\boldsymbol{o}, \boldsymbol{\mu}_{im}, \boldsymbol{U}_{im}], \qquad 1 \leq i \leq N. \tag{2.7}$$

In Equation 2.7 the expression c_{im} is the mixture coefficient for mixture m in state i and $\mathfrak{N}$ can be a Gaussian function with mean vector $\boldsymbol{\mu}_{im}$ and covariance matrix $\boldsymbol{U}_{im}$ for mixture m in state i. Although Gaussian functions are often applied, other functions can be used and are sometimes more appropriate. For instance for modelling observations with a strictly positive value range an exponential decay function can be applied. Generally speaking, any log-concave or elliptically symmetric density can be used (Liporace, 1982) as long as the mixture satisfies the stochastic constraint

$$\sum_{m=1}^{M} c_{im} = 1, \qquad 1 \leq i \leq N \tag{2.8a}$$

$$c_{im} \geq 0, \qquad 1 \leq i \leq N, \;\; 1 \leq m \leq M \tag{2.8b}$$

giving a normalised probability density function with

$$\int_{-\infty}^{\infty} b_i(\boldsymbol{x})\, d\boldsymbol{x} = 1, \qquad 1 \leq i \leq N. \tag{2.9}$$

The concept of continuous emission HMMs can also be illustrated using a simple example. Imagining a situation where an observer is making repeated invocations of a website trying to infer the routing of his or her requests based on the response latency. Furthermore, the world wide web is assumed to only feature a finite number of static routings from the observer to the target website, while each routing has its specific probability distribution of connection latency. For every invocation of the target site a routing is chosen alternately following a hidden pattern. Translating this setup to a continuous HMM, the hidden process X is the current routing for accessing the page, $E \in [0; \infty)$ is the experienced latency and the emission matrix B describes each routing's latency probability distribution. If B is known the observer can now make inferences about the hidden sequence X using latency measurement E.

2.2.2 HMM-Related Inference Problems

According to the highly esteemed work by Rabiner (1989) a number of common inference problems are associated with hidden Markov models. The following paragraphs give an overview of the inference problems and their solutions relevant to a wide range of applications:

Learning of Model Parameters

Given a set of observation sequences, the foremost problem is to determine the model's parameters best fitting these observations. Specifically, the question is how to set the parameters to maximise $P(\mathcal{O}|\theta)$ with $\theta = (S, E, A, B, \pi)$ and $O_n \in \mathcal{O} = (o_1, o_2, \ldots, o_t), o_n \in E$. Unfortunately, this essential problem is also the most difficult of the inference problems and no analytical solution exists which solves the problem optimally (Rabiner, 1989, page 8). Instead, practitioners have to rely on iterative procedures.

Most prominent the *Baum-Welch algorithm* (Baum et al., 1970) is used which gradually tunes model parameters, resulting in a maximum likelihood estimation for a given set of observation sequences. However, this procedure does not necessarily converge to the globally optimal solution. Instead, depending on the complexity of the problem space, the Baum-Welch algorithm may only yield a local optimum. Because the algorithm's outcome is largely influenced by the initial starting values for the iterative optimisation procedure—which have to be guessed in many application scenarios— practitioners generally choose many different start value configurations, run the optimisation repeatedly and afterwards chose the best performing model(s) for further analysis according to an application specific performance metric.

Determining Sequence Probability Given a Model

This problem concerns the calculation of occurrence probability $\mathbb{P}(O|\theta)$ of any given sequence of observations $O = (o_1, o_2, \ldots, o_t), o_n \in E$ with respect to a fully specified model $\theta = (S, E, A, B, \pi)$. Intuitively this problem maps to the question of how good an observations sequence fits to a given model. An efficient solution to finding sequence probability is using the forward part of the *forward-backward algorithm* (Baum and Eagon, 1967; Baum and Sell, 1968). The most important application of this forward algorithm is determining the best fitting model from a set of candidate models given an observation sequence, which can be viewed as a classification task.

Calculating Probability of Latent Variables

Based on a sequence of observations $O = (o_1, o_2, \ldots, o_t), o_n \in E$ and a model $\theta = (S, E, A, B, \pi)$ this categorical problem deals with finding the best possible explanation for O in terms of a sequence of hidden states $S = (s_1, s_2, \ldots, s_t)$. Unlike the former problem there exists no exact solution. Which hidden state sequence best explains the observations rather is dependent on the selected optimality criterion, while the actual choice of criterion depends on the application domain. A possible optimality criterium could be, for example, finding the states that are individually the most likely without regarding the sequence of states, maximising the number of correctly guessed states. However, for non-ergodic models[1] this procedure may produce invalid state sequences. While in theory one could overcome this problem by considering state pairs, triples and so on, the most common optimality criterion is finding the single best fitting state sequence for the whole sequence of observations. This procedure is called the *Viterbi* algorithm (Forney, 1973; Viterbi, 1967). Finding the best possible explanation is useful whenever hidden states can be mapped reasonably well to real properties of the system at question and additional knowledge about these properties is sought after.

Choice of Model Properties

When it comes to applying HMMs in practice, several other design choices remain to be made. Given the nature of the data or problem to be modelled, some of these choices may be dictated while others remain at the discretion of the practitioner. A typical parameter dictated by the problem type is what type of model is best to be used. In many applications related to speech recognition so called *left-to-right models* are dominant, where a hidden state can not be revisited once it has been left. Such models often also include an explicit sequence ending state. The most versatile structure of the hidden process is used in *ergodic* models, where every hidden state is reachable from every other state in the model at every time step. Combinations of these model types are also applicable in certain domains.

Furthermore, the choice of observation symbol is strongly influenced by the data itself, where discrete observations naturally lead to discrete HMMs. When dealing with continuous signals other options arise to practitioners. Most straightforward is the application of a continuous observation distribution (e.g. Gaussian function) for single feature continuous observations. When dealing with richer feature vectors, however, covariances between features have to be regarded as well, quickly leading to more complex observation functions. Sometimes it may also be ap-

[1] In a non-ergodic model not every state is reachable form every other hidden state, i.e. there exits a transition probability $a_{ij} \in A = 0 | i \neq j$

propriate to discretise continuous features, sacrificing a model's accuracy for its simplicity. It is also worth noting that there are no straightforward mechanisms for feature selection in HMMs, as they are commonly available for many other machine learning techniques.

Additionally to above mentioned aspects, there is also no general solution for finding the correct size of a model in terms of the hidden state count. While there are some domains where this choice can easily be deducted from the problem (DNA analysis is a good example), in other cases there may not even be a *correct* model size or the apparently most suitable model size is dependent on the performance metric used. As it is also true for most of the afore mentioned design choices, practitioners have to rely on experimentation and explore potentially many different configurations for any given dataset, iteratively improving the generated models. Generally it can be said that the more domain knowledge is available the easier some or most of the afore mentioned design decision become when applying HMMs.

Finally, it is also worth mentioning that depending on the application sometimes practitioners choose to relax some of the hidden Markov model constraints to improve modelling results. Similar as for the simpler Markov models, there exists a number of extended HMM definitions that can be used for this purpose. Notable examples in the context of modelling search are for example variable-order HMMs, which not only are able to model higher-order dependencies but also allow to modify the order based on the observed sequences. These models are for example used by Cao et al. (2009) to capture the context of users' queries based on a very large scale search engine log. While their work focuses on the challenges arising from the sheer size of their dataset, the authors also demonstrate how variable-order HMMs can be used to generate URL recommendations and query suggestions.

A further HMM variant are partially observable Markov models, where hidden state transitions do not necessarily produce observable results. Wang et al. (2010) use partially observable Markov models to analyse a search engine log, accommodating for the fact that in their case no traceable record is left in the log when a result item is read. He and Wang (2011) further extend on Wang et al.'s model of browsing and clicking behaviour on a search engine result page by also including a temporal component in their models of user behaviour.

2.3 Multi-Space Hidden Markov Models

In the previous sections two major categories for HMMs were introduced based on either discrete or continuous definition of emission probability functions. How-

ever, both definitions suffer from the fundamental limitation that they assume either *all* features to be discrete or *all* features to be continuous within any individual model. Since this is also the case for the variable-order HMMs as well as the partially observable Markov model, none of the definitions given so far is suitable for applications where discrete and continuous features occur simultaneously or alternately in an observation sequence. Although an elaborate setup of hidden states and transitions can be used to overcome this limitation in domains where extensive knowledge about the subject is available, a general-purpose solution to this problem requires a more comprehensive formalisation of HMMs.

Multi-space hidden Markov models introduced by Tokuda et al. (1999) use multi-space probability density functions which are in fact capable of unifying discrete and continuous features into a joint model in the most general setting. The authors originally applied this type of HMMs in a series of experiments related to natural speech recognition and synthesis, yielding superior results compared to related approaches employing conventional HMMs (Tokuda et al., 2002, 2013).

In a multi-space HMM the emission functions are basically m-mixture models, where each $m \in M$ of the mixtures can be "activated" arbitrarily given each individual observation. Note that in the definition in Section 2.2.1 the m-mixture component is assumed to be used only in its entirety. This relaxation of assumptions makes multi-space HMMs more flexible than both definitions of HMMs given above. Furthermore, discrete and continuous (conventional) HMMs are special degenerate cases of multi-space HMMs, which ensures that above mentioned standard solutions to HMM-related inference problems are still applicable with small modifications (for details refer to Section 3.2).

Downward compatibility also becomes clear when looking at the formal derivation of multi-space HMMs, where the multi-space observation probability distributions have a sample space Ω containing G spaces with the property

$$\Omega = \bigcup_{g=1}^{G} \Omega_g. \tag{2.10}$$

Furthermore, each Ω_g describes an n-dimensional real space R^{n_g} where g is the space index. Associated with each space is a probability value w_g, where

$$\sum_{g=1}^{G} w_g = 1 \tag{2.11}$$

additionally to an observation probability density function of the form

$$N_g(x), x \in \mathbb{R}^n, \text{ where } \int_{-\infty}^{\infty} N_g(\boldsymbol{x}) d\boldsymbol{x} = 1. \tag{2.12}$$

Consequently, a discrete emission HMM is created by setting dimensionality $n_g = 0$ for all spaces $\Omega_g, g \in G$ comprising the set of unique categorical emissions. Each space is parametrised with a specific weight w_g, which is determined by the probability of observing the respective categorical emission. Additionally, multi-space observation probability distributions spanning the entire sample space Ω are assigned to all model states, yielding a setup equivalent to discrete emission probability as introduced in Equation 2.6.

When taking a closer look at Equation 2.12, it also becomes apparent how multi-space HMMs relate to HMMs using continuous emission functions. Specifically, by setting spaces' dimensionality $n = 1$, each space's observation probability density function simplifies to the setup described in Equations 2.7–2.9. If it is furthermore assumed that every observation always contains all space indices $g \in G$, a multi-space HMM consisting entirely of one-dimensional spaces is equivalent to a continuous emission HMM as described in Section 2.2.1.

However, the fundamental strength of multi-space HMMs compared to discrete and continuous HMMs is not only their usage of higher-order probability functions. Rather it is the richer definition of the observable events $e_i \in E$ generated by the random process $\{Y_t\}_{t\in\mathbb{N}}$ that leads to their superior expressiveness. In contrast to conventional HMMs, in the multi-space definition each observation event is given by a random vector $\boldsymbol{o} = (X, \boldsymbol{x})$. The vector consists of a continuous random variable $\boldsymbol{x} \in \mathbb{R}^n$ and the set of space indices X where all spaces $\chi \in X$ must be n-dimensional. On the other hand, not all n-dimensional spaces have to be included in X. Intuitively mapping this to the m-mixture observation function for continuous HMMs (Equation 2.7), the multi-space approach allows to specify which components of the m-mixture are to be activated for each individual observation. However, it is important to note that not only the random variable $\boldsymbol{x}$ but also the space activation vector X is a random variable and its properties are determined by the modelling subject.

Consequently, multi-space observation probability of $\boldsymbol{o}$ is defined as

$$b(\boldsymbol{o}) = \sum_{g \in \boldsymbol{X}} w_g N_g(\boldsymbol{x}) \tag{2.13}$$

while the hidden states, the hidden state transition matrix A and the starting state probability vector π are defined analogously to a conventional HMMs.

The general strength of multi-space HMMs can best be illustrated using a notional

example for how user behaviour could be modelled on the basis of a search engine transaction log. Additionally to the information which categorical actions were conducted by the user, it is assumed that the log contains the actions' durations as well. Furthermore, a number of additional features are made accessible by the search engine log, for example, the length of submitted queries, the ranking of result sets and the documents' length as well as their (binary) relevance value with respect to a query.

What makes this HMM variant powerful in the given scenario is their hidden states' multi-space emission functions which allow to describe all available features in a unified model. For example, in a multi-space HMM it is possible to model a click on a result item as observation $\boldsymbol{o} = (\langle result, rank, relevance \rangle, \langle 5, 2, 1 \rangle)$, indicating that it took the user five seconds to click on the second result item, which was relevant to the query.

To achieve this sophisticated model of user behaviour, the hidden states emission functions need to be initialised as follows: First, a set of spaces is created, where each available feature is represented by one or more spaces. For example, to model the actions' times, one approach is to use a single space for every possible user action. The actual actions' times are then modelled using an appropriate probability density function, for example, an exponential decay function. The remaining features are also modelled in their own respective spaces, which are then initialised with suitable functions describing the features' observable values.

Apart from distinguishing between discrete and continuous features when chosen the individual spaces' functions, it is also important to consider the range (the support) and likely distribution of the features' values. For example, to model query length, an exponential probability mass function is appropriate since from the literature is is known that longer queries become increasingly less likely. It is worth noting that query length is a discrete feature, therefore, a probability mass function is used instead of a probability density. While document length could be modelled similarly to query length, documents' relevance could be modelled as a Bernoulli distribution (binary relevance assumption) or using a beta distribution when relevance is given on a continuous scale.

In the final multi-space model each spaces' emission function is activated only if the respective space's index is given in an observation $\boldsymbol{o}$, which allows to model the situation illustrated above without the usage of any heuristic assumptions.

2.4 Markov Decision Processes

Additionally to the modelling techniques based on Markov chains which assume an autonomous system, another category of models follows the controlled system

assumption. In a controlled system the random process' response is stimulated by an action chosen by a *decision maker*. If the random process is markovian and fully observable both components constitute a Markov decision processes (MDPs). A MDP is a direct extension of a Markov chain introducing *choice* and *reward* as additional concepts. Consequently, a MDP offering only a single possible action and no reward reduces to a Markov chain.

Formally, a Markov decision process consists of a quintuple $(S, A_s, P_a, R_a, \gamma)$, where S is the set of possible system states and A_s is the set of possible actions in state $s \in S$. The actions' probabilities are given by P_a, where

$$P_a(s, s') = \mathbb{P}(s_{t+1} = s' | s_t = s, a_t = a) \tag{2.14}$$

indicates the probability of changing to state s' at time $t + 1$ when action a is executed at time t. Furthermore, $R_a(s, s')$ denotes the reward for a state transition from state s to state s' due to an action a and $\gamma \in [0, 1]$ is a factor that discounts future actions' rewards compared to immediate rewards.

Given the fully specified quintuple, the general idea of applying MDPs is to find an optimal decision path for the decision maker, maximising the cumulative discounted reward gained by choosing specific actions along the path. Since the model follows the Markov assumption, a general solution to the problem can be expressed in terms of the current system state. This allows efficient calculation of the optimal decision path over a potentially unlimited horizon by using e.g. dynamic programming to solve the Bellman optimality equation (Bellman, 1954).

MDPs were introduced in the same era that also spawned development of HMMs mainly by Bellman (1957). Howard's (1960) book on MDPs led to their earliest applications, focussing on industry related domains and economics. Another more recent and extensive summary on the topic and its variations is also given by Puterman (1990, 2014).

2.5 Partially Observable Markov Decision Processes

Similar to how HMMs extend Markov chains for modelling problems with unknown components, partially observable Markov decision processes (POMDPs) extend MDPs by introducing hidden states. The general idea of modelling an agent's decision process remains unchanged compared to MDPs but since the underlying system state is not observable by the agent, POMDPs also include a *belief state* as an additional component. The belief state is a probability distribution over the set of possible states indicating the likelihood of specific system

states at the current time step. It is maintained by the agent and updated based on the observations made, the observation probabilities and the underlying MDP.

Formally, a POMDP is a septuple $(S, A_s, P_a, E, O, R_a, \gamma)$, where S, A_s, P_a, R_a and γ are defined analogously to fully observable Markov decision processes. Additionally, the system is assumed to be in a state s and after the decision maker takes action a it transitions to a new state s' while emitting an observable symbol $e \in E$, where E is the set of possible observations. The final component in a POMDP is the set of conditional observation probabilities O, where $o_a \in O = \mathbb{P}(e|s', a)$ denotes the probability of observing e after action a triggered the system to change to state s'.

This type of model constitutes the most general approach based of Markov chains, unifying the idea of hidden states with a cost and reward driven decision making agent. Therefore, they are applicable to a wide range of problems related to sequential data. The general idea of POMDPs was introduced in the fundamental work of Åström (1965) and later extended in various domains, e.g. by Kaelbling et al. (1998) in the artificial intelligence community. Cassandra (1998) also provides an extensive survey of POMDPs in practice, ranging from industrial, business, military and social to scientific applications.

More recently, Luo et al. (2014) show how POMDP can be used to model session search as a stochastic game. The authors introduce the user and search engine agent as components of the system which are working together cooperatively to maximise a long term reward. In a subsequent work, Luo et al. (2015) further investigate on the complex problem of choosing appropriate POMDP parameters for a specific dataset. In this work the authors give advice on how to design states (fixed vs. variable number), select sub-sets of the available search engine actions to include in the model and on usage of possible reward functions using explicit or implicit feedback mechanisms.

Jin et al. (2013) use POMDPs for document re-ranking in complex search sessions. In their approach user feedback on the first result page is used to generate a personalised ranking on following pages, striving to optimise the trade-off between exploitation and exploration in a result set. Zhang et al. (2014) also use POMDPs for document re-ranking based on a search engine log. Interestingly, their approach manages to outperform commercial search engine ranking without considering query or document content, purely based on the users' interaction patterns. Finally, Yang et al. (2018) review the literature related to modelling session search using POMDPs to derive practical design recommendations for implementing this method.

Although POMDPs are a powerful and very general tool for sequence modelling they are not applicable in every situation. Specifically, POMDPs require the practitioner to explicitly specify a reward function which might not be possible

in every situation (Yang et al., 2018). Furthermore, exactly finding the optimal policy in POMDP problems is computationally intractable (Hauskrecht, 2000), requiring practitioners to use heuristic solutions or sampling techniques. However, not all problems feature characteristics that can be exploited to ease computational complexity, in which case it can be more appropriate to use other modelling techniques.

2.6 Conditional Random Fields

Additionally to the Markov chain based approaches, in the following section *conditional random fields* (CRFs) are introduced as an alternative sequence modelling technique. CRFs were first described by Lafferty et al. (2001) in the context of natural language processing problems such as part of speech tagging. Additionally, they have also been used in other tasks, e.g. biomedical named entity recognition (Settles, 2004) or (moving) image segmentation (Wang et al., 2006; Wojek and Schiele, 2008). CRFs were also applied to model search behaviour, for example by Ageev et al. (2011), who also identified the need to analyse actual log data on a larger scale. However, instead of striving to create a holistic model of search, the authors focus on proposing a categorisation of users' actions by successfulness.

Out of the previously introduced modelling approaches, CRFs are closest related to hidden Markov models as it also becomes apparent by the similar practical applications. In fact, CRFs can be viewed as an extension of HMMs that relaxes some of the fixity assumptions about the probability distributions made by HMMs at the cost of an overall higher conceptual complexity. Vice versa, a HMM can be viewed as a linear-chain CRF with a specific and constant function for state transitions and observation probabilities. In contrast to Markov models, which even for higher-order variants can only model short dependencies, CRFs have the whole input sequence available for making predictions at any time step. Learning of linear-chain CRFs' parameters can be achieved by using variants of the forward-backward and Viterbi algorithms. However, these variants have a higher computational complexity compared to their HMM counterparts (Sutton and McCallum, 2006, Page 110ff).

Although CRF and HMM approaches are similar in application as well as in the inference and estimation algorithms used, there is a fundamental difference: Principally, CRFs belong to the discriminative model family (together with e.g. maximum entropy models) which estimate the probability of a label given an observation $\mathbb{P}(Y|X = x)$. In contrast, HMMs and their extensions are generative models, which means they model observed as well as hidden variables in a joint probability distribution $\mathbb{P}(X, Y)$.

2.7 Deep Learning

In recent years deep neural networks have gained a substantial boost in popularity in the machine learning community. The basic ideas behind neural networks are far from new. Rosenblatt (1957) for instance described the concept of a *perceptron* as early as 1957, which decades later turned out to be the foundation for *convolutional neural networks*. However, the remarkable rise in popularity of neural networks—nowadays often just being referred to as *deep learning*—was only made possible by advances in computer technology and hardware optimisation in recent years. Since then deep learning has been applied to numerous problems from a wide variety of domains, ranging from the automated generation of images (Radford et al., 2015) to diagnosing skin cancer (Esteva et al., 2017).

Deep learning can also be applied to sequence modelling by using long short-term memory networks (LSTM) introduced by Hochreiter and Schmidhuber (1997). These networks use special neurons—memory cells and gate units—to selectively remember patterns for a variable length of time. Therefore, this models are well suited for tasks were the distance between important events is unknown a priori.

High impact applications of LSTM are for instance the work by Capes et al. (2017) who have successfully enhanced text-to-speech quality of Apple's electronic personal assistant Siri. Goyal et al. (2018) showed how special forms of LSTM can be used to transfer existing knowledge to low resource domains in the Amazon Alexa system. Furthermore, Borisov et al. (2016) use LSTM to model users' browsing behaviour in web search and apply their models to predict users' next clicks on results as well as to re-rank the result set. Sequences of user actions were also used by Twardowski (2016) to built a session-aware recommender system that does not require explicit information on the user. Kochkina et al. (2017) use LSTM to model the conversational structure in tweet sequences to classify the tweets' stances.

Despite the fact that deep learning and particularly LSTM in principle could also be applied to the problems of session search and rumour veracity modelling, pursuing this direction of research is out of the scope of this thesis and left for future work.

2.8 Technology Selection

In the following section the previously introduced modelling techniques are compared with respect to their strengths, weaknesses and prerequisites regarding problem and source data types. Based on these considerations conclusions for choosing the appropriate technique for this thesis are drawn.

The first consideration was briefly touched on before and is concerning whether a deep learning or a conventional machine learning approach is preferred. For that matter, it should be kept in mind that the overall goal of this work is not only result oriented. Instead, one of the central desired properties of the modelling framework is that the models' structure should also be transparent, which allows to analytically investigate model parameters and to use these insights to predict future events. Considering this, it becomes apparent that the black box character of deep learning algorithms is inappropriate.

Therefore, focussing on conventional sequence modelling techniques, two major categories of models can be determined: Generative (Markov model family) and discriminative models (CRFs). Out of those, only the generative approach has the benefit of being able to generate new data based on previously observed data. Especially for advanced tasks in user modelling concerning session search, such generative algorithms can become very useful, for example, for simulating user behaviour for system evaluation purposes or for applying user guidance based on predicted future user actions.

Furthermore, generative models have additional properties which are preferable in many situations compared to discriminative models. While the former can generally also work very well with unlabelled training data, the latter require large amounts of labelled training data (Bernardo et al., 2007). Depending on the task, these labels may be unavailable or costly to acquire. Furthermore, in comparative studies it has been noted that generative models need less training data to realise their optimal performance in a specific task (Ng and Jordan, 2002). In contrast, discriminative models can have a lower asymptotic error rate in some cases.

Focussing on HMMs and CRFs as concrete implementations of both principal modelling categories, a number of further observations are made in the literature. In Lafferty et al.'s (2001) fundamental paper on CRFs the authors motivate their research by stating that HMMs have difficulties with modelling multiple interacting features and long range dependencies—when being restricted to first-order HMMs. In contrast, CRFs were shown to perform better if higher-order dependencies exist in data. However, this specific experiment is based on synthetic data only. Moreover, Lafferty et al. also state that training a CRF is in general slower compared to HMMs, especially when longer input sequences are used, as the complexity of the CRF grows proportional with the length of the input. The authors focus their considerations mainly on the domain of natural language processing. Complexity growth may not be a significant problem in this research area because of the natural length limitation of spoken or written sentences. When considering a general purpose framework, however, computational complexity of CRFs must also be regarded, especially when dealing with a large number of training instances or long sequences (Sutton and McCallum, 2006).

Feng et al. (2006) also compared CRFs and HMMs, in this case with respect to the task of handwriting recognition based on discrete features only. The authors argue that although CRFs perform better at this particular task, their application is also associated with certain difficulties. In fact, the authors state that CRFs have a large state space requiring comparably more parameters to be estimated, which results in a high computational complexity in practice. Furthermore, the authors found that a direct usage of continuous features is problematic in the context of CRFs.

Based on the considerations made above, CRFs are less well-suited for the tasks at hand. Instead a generative model from the family of Markov models is preferable because of its higher flexibility and ease of use. As also shown in Table 2.1, variants of Markov models are capable of describing partially and fully observable systems. Since the problems formulated in the introductory part of this thesis also feature hidden variables, only the latter systems are applicable here, namely HMMs and POMDPs.

Out of these two models POMDPs are more powerful since they define a controlled system. However, higher expressiveness also increases the models' computational complexity substantially. Additionally, POMDPs require a comparatively large amount of training data, as it is also pointed out in Cassandra's (1998) survey on POMDP applications, where the author states that POMDPs are data intensive since they need a data point for every combination over all possible states, observations, actions and rewards.

Fully defining a POMDP in particular also requires the specification of a reward function. However, depending on the application such a function may be difficult to find without using strong assumptions or heuristics. In other cases, the concept of rewards may not be applicable at all, i.e. when there is no controlling agent present in the system.

Finally, one of the goals of this thesis is also to provide a flexible general purpose modelling framework which is applicable to a wide range of scenarios. HMMs have the advantage that they can be adapted to different task complexity levels as well as input data types very easily, i.e. by using discrete, continuous or multi-space observations. When striving to build a general purpose framework, HMMs are the most flexible technique to use while still being reasonable powerful.

After thoroughly motivating the technology choice, in the next chapter the implementation of a HMM-based general purpose sequence modelling framework is presented.

CHAPTER 3

Implementing a Framework for Multi-Space HMMs

In the previous chapter a range of principle approaches for modelling temporally ordered sequences were presented. When balancing pros and cons of the discussed methods, it becomes clear that hidden Markov models are best suited to achieve the research goals outlined in Chapter 1.2. Since HMMs are a popular sequence modelling technique that has been introduced decades ago, evidently there exits a wide range of implementations written in various scripting and programming languages, for example in R[1], Python[2] or Matlab[3].

However, these implementations are generally tuned towards ease of use, since they are often specifically targeted at users with limited background knowledge of machine learning techniques. Tools prioritising simplicity often have deficiencies in flexibility and in this case common HMM implementations impose undesired restrictions on the models' structure, the parameter tuning process or the possible value range of the observation alphabet. However, another reason ultimately disqualifies available implementations from out of the box usage for user behaviour modelling in the general case: Their inability to simultaneously process observations on discrete and continuous scale.

Especially, when specific knowledge about the underlying hidden process is limited in the respective domain, joint modelling of all available observation features

[1] `https://www.rdocumentation.org/packages/HMM/versions/1.00`—last accessed 24.10.18
[2] `https://hmmlearn.readthedocs.io/en/latest/index.html`—last accessed 24.10.18
[3] `https://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html`—last accessed 24.10.18

is essential to capturing the temporal component of user behaviour. Multi-space HMMs provide the extended theoretical background to realise this joint modelling. Nevertheless, as of today, there exists no publicly available general purpose implementation of multi-space probability density functions.

The work presented in this chapter strives to overcome the deficiencies in flexibility and modelling power present in prior HMM implementations. Hereby, the foremost goal is not only to create an implementation of the multi-space modelling paradigm, but to integrate it into a joint general purpose framework applicable for creation of the most common HMM variants. Additionally, the framework is designed so that full control over all involved parameters is guaranteed at any time, both with respect to the generated models' structure as well as to the actual iterative tuning procedure.

The benefit of such a flexible framework is at least twofold: On the one hand, high customisability of HMMs constitutes one of the key arguments for using these over other modelling techniques, especially when dealing with user behavioural data. In certain cases it may be necessary to encode domain knowledge into a model's hidden process, which for instance can be achieved by targeted specification of individual parameters in the state transition matrix. On the other hand, it can also be important to consider different modelling variants—e.g. discrete vs. multi-space—given a particular dataset or a research question, especially since HMMs are not capable of performing automatic feature selection. Based on the joint framework presented here, different feature sets can easily be compared in terms of their performance once a dataset has been made available to the framework.

In the next section the modelling framework's general architecture is presented briefly, before describing the implementation of the multi-space observations probability density functions. Afterwards, the necessary steps to adapt the framework to a particular use case are discussed.

3.1 Java HMM Library for Basic Algorithms

The actual implementation of the framework is written in the concurrent and object-oriented programming language Java. Apart from the aforementioned implementations of HMMs written in various other languages, there exists also a small number of libraries written in Java, albeit these principally also exhibit the same shortcomings as discussed above. Nevertheless, the HMM framework presented here is partially based on the low-level library *Jahmm*[4], in particular by using the library's implementations of core concepts related to discrete and continuous HMMs, for example the Baum-Welch and Viterbi algorithms.

[4] `https://github.com/KommuSoft/jahmm`—last accessed 24.10.2018

The library is also written generically, thus ensuring its suitability for a broad range of problems, as opposed to many other solutions which are targeted at specific use cases, for example bioinformatics or speech recognition. Furthermore, Jahmm is licensed under GNU General Public License[5], which allows usage and modification of source code. Since the library's active development came to a halt in late 2014, its source code was directly included in the framework, instead of using the binary files. This allows full control over the code base, which was thoroughly tested to identify and resolve any open programming issues.

In certain user behaviour modelling tasks no adequate knowledge about the hidden process is available, for example when modelling hidden temporal patterns in stance distributions to determine rumours' veracity. To explore how the enhanced expressiveness of multi-space HMMs can be exploited in these tasks, multi-space probability density functions are integrated into the common software architecture shared with discrete and continuous probability (density) functions. As a result the implementations of central algorithms like such as Baum-Welch parameter tuning, Forward-Backward procedure and Viterbi sequence probability estimation can be reused for all HMM variants, independently of their respective observation values' ranges. Considering that discrete as well as continuous models are special cases of the more general multi-space modelling variant, this software engineering design choice is appropriate also from a mathematical point of view.

Especially, due to the shared software architecture all modelling variants can be used interchangeably with minimal effort, since hidden state transition matrix A, the start vector π as well as the general parameter tuning processing logic remain unchanged regardless of the chosen modelling variant. Consequently, when applying the framework in practice the models' actual type is determined by initialising the models' hidden states with the desired probability (density) functions (discrete, continuous or multi-space) and conducting the training procedure by feeding appropriately preprocessed observation sequences to the tuning algorithm.

3.2 Multi-Space Observation Probability Density Functions

To concept of multi-space observation probability density functions was adopted from Tokuda et al. (1999), who proposed it to improve the quality of speech recognition as well as artificial speech generation. In their models of pitch patterns in natural speech they describe observation sequences as a joint signal of frequency readings represented by a one-dimensional continuous variable for the

[5] `https://www.gnu.org/licenses/gpl-3.0.en.html`—last accessed 24.10.2018

voiced regions of speech and a discrete symbol that indicates observation of *unvoiced* regions.

This particular instantiation of multi-space observations can be transferred to modelling temporal patterns in user behaviour, also making use of the joint modelling of discrete and continuous signals. In this case, each observation is considered to be a random variable $\mathbf{o} = (o_s, o_x)$ consisting of two components, the discrete user action s and the action's time x, where x can be the duration of the action or the distance of time to the preceding action.

In a multi-space HMM based on this definition of observation signals, each of the G items in the alphabet of observable user actions Ω spans its own space $\Omega_g \in \Omega$. Furthermore, for each Ω_g all observed action occurrences' times x are described by a probability density function

$$N_g(x), x \in \mathbb{R}^1, \text{ where } \int N_g(x)dx = 1. \tag{3.1}$$

The individual spaces' probability density functions can be aggregated to form a multi-space observation probability density function, which is used to initialise the hidden states. Consequently, a multi-space models' emission probability matrix B can be fully specified as a combination of all states' multi-space observation functions. It is worth noting that generally not every $\Omega_g \in \Omega$ needs to be defined for every hidden state in the model. If any space is undefined in a specific state $n \in N$, this indicates that the respective user action can not be observed in n.

Two concrete implementations of multi-space observation probability density functions were created as part of the framework, one based on Gaussian distributions and the other based on exponential decay distributions. These functions are commonly used for modelling continuous observations using hidden Markov models and appropriate for many types of statistical phenomena.

Strictly speaking, for modelling user actions' times only the usage of exponential decay functions is mathematically sound, since their support is defined as $x \in [0, \infty)$ and times can not be negative. However, in practice both functions can be used interchangeably with only minimal effect on the modelling outcome. Furthermore, the multi-space principle can be applied to continuous features unrelated to time as well, which may have different value ranges. Consequently, both functions will be discussed below.

In a multi-space HMM whose spaces are initialised using exponential decay functions the probability of making any observation $\mathbf{o} = (o_s, o_x)$ is given by the expression

$$b(\mathbf{o}) = \mathbb{P}(o_s, o_x) = w_s \lambda e^{-\lambda x} \tag{3.2}$$

where w_s is the weight of the respective space s observed in $\mathbf{o}$ and λ is the function's decay rate. Similar, the probability of any observation $\mathbf{o}$ can also be expressed by using a Gaussian function of the form

$$b(\mathbf{o}) = \mathbb{P}(o_s, o_x) = w_s \frac{e^{-(x-\mu)^2/2\sigma^2}}{\sqrt{2\pi\sigma^2}} \tag{3.3}$$

where μ is the function's mean and σ^2 is the function's variance.

The emission probability matrix B of multi-space HMMs is tuned according to the training data alongside with transition matrix A and start vector π, following the Baum-Welch algorithm essentially analogous to conventional HMMs. However, the algorithm needs to be extended to account for the fact that the observations in the training data O need to be categorised by their respective spaces Ω_g before tuning the spaces' probability density functions $N_g(x)$. Consequently, tuning of A, B and π in the multi-space setting is conducted following the algorithm outlined in pseudo code below.

Listing 3.1: Tuning model parameters in multi-space HMMs

```
1  input initial model θ, observations O
2  output tuned model θ̄
3  while convergence has not been reached
4      fill forward array α and backward array β
5      foreach state n in θ
6          foreach space Ω_g known in n
7              extract O_{Ω_g} = {o ∈ O}, where Ω_g ∈ o
8              fill scaling array γ
9              calculate scaled μ and σ² using γ and O_{Ω_g}
10             update N_g(x) in Ω_g according to scaled μ and σ²
11         update multi−space probability function in n
12 end while
```

The α and β arrays calculated in Line 4 of Listing 3.1 contain all probabilities of partial observation sequences in forward (Equation 3.4) and backward (Equation 3.5) direction, given that q_t is the state of the hidden process at step t and that the sequence has a length of T steps.

$$\alpha_t(i) = \mathbb{P}(o_1, o_2, \ldots, o_t, q_t = s_i|\theta) \tag{3.4}$$

$$\beta_t(i) = \mathbb{P}(o_{t+1}, o_{t+2}, \ldots, o_T|q_t = s_i, \theta) \tag{3.5}$$

Consequently, the arrays can be calculated following the same iterative procedure as for conventional HMMs detailed in Equations 3.6 and 3.7 (Rabiner, 1989). Note that the actual implementations use a scaling procedure to prevent arithmetic underflows, which is left out from the equations for the sake of readability.

Forward

$$\begin{aligned}
&\textit{Initialisation}: \\
&\qquad \alpha_1(i) = \pi_i b_i(o_1), && 1 \leq i \leq N \\
&\textit{Induction}: \\
&\qquad \alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1}), && 1 \leq t \leq T-1 \\
& && 1 \leq j \leq N
\end{aligned} \tag{3.6}$$

Backward

$$\begin{aligned}
&\textit{Initialisation}: \\
&\qquad \beta_T(i) = 1, && 1 \leq i \leq N \\
&\textit{Induction}: \\
&\qquad \beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), && t = T-1, T-2, \ldots, 1 \\
& && 1 \leq i \leq N
\end{aligned} \tag{3.7}$$

In Line 8 of Listing 3.1 the α and β arrays are used to estimate the posterior probabilities of being in state i at time t following Equation 3.8.

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \tag{3.8}$$

Finally, the scaling factors γ are used in Line 9 to re-estimate the probability density function's parameters in the multi-space setting as described by Tokuda et al. (2002). The update procedure outlined in Lines 9-10 of Listing 3.1 illustrates usage of Gaussian observation probability density functions, but other functions can be used similarly to best fit the data. For example, when an exponential decay function is assumed instead, its parameters can be derived directly from the observation values' scaled mean by setting the decay rate $\lambda = 1/\mu$. After updating all probability functions, the process is repeated until an acceptable degree of convergence or a preset maximum of iterations has been reached.

3.2.1 Generalisation of the Probability Functions

In the setting described in the previous section it is assumed that all observations in the multi-space HMM contain exactly one space index, which is denoting the action executed by the user. The actions' times are consequently modelled solely using the respective space's observation probability density function.

At first glance it may seem that this setting could equivalently be modelling using much simpler continuous HMM parametrised with multi-Gaussian (or exponential) emission functions, allowing to omit the additional effort to model the spaces. In a multi-Gaussian HMM, each observation $\mathbf{o}$ would be defined as a vector of dimensionality n, where n equals the size of the observable user action alphabet E. At any time step t exactly one element of the vector takes a value $x \in \mathbb{R}_{>0}$ to indicate that a user conducted the respective action, while all other elements of the vector are set to $x = 0$.

However, the key factor is that this would still imply that all actions were performed at time t—though only one action has time larger than zero. In comparison, a multi-space HMM allows to truly model that an action was not observed at time t since all actions' spaces can be addressed separately. Therefore, the multi-space HMM is more general than a continuous HMM using multi-Gaussian emissions and the preferred modelling variant in many cases.

Moreover, the multi-space HMM framework has the advantage that it can in principle be generalised even further to model events more complex than the examples discussed so far. Conceptually, observations do not need to be constraint to a single space at any time step t. Instead an observation at time t can also be given as $\mathbf{o}_t = (S(o), x)$, where $S(o)$ is a vector specifying an arbitrary set of space indices $g \in G$ and x is a continuous observation $x \in \mathbb{R}$. The parameter

tuning procedure described in Listing 3.1 can unalteredly be applied in this circumstance when the scaling parameter is generalised to model the observations' distribution across the spaces. Equation 3.9 details how the extended scaling parameter $\gamma_t(i,h)$ can be calculated for space h of state i at time step t using space weight w_{ih} and the probability density function $N_{ih}(x)$.

$$\gamma_t(i,h) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N}\alpha_t(j)\beta_t(j)} \cdot \frac{w_{ih}N_{ih}(x)}{\sum_{g\in S(o)} w_{ig}N_{ig}(x)} \tag{3.9}$$

Consequently, the framework can also be used to model user behaviour in applications where only partial information about the performed user actions is available. For example, when data is acquired using a noisy channel, a particular element in a sequence of user actions may not be recorded correctly. In this case this action's time can be deducted from the preceding and subsequent user actions. However, since it is unknown which user action was originally performed, the action's time has to be modelled using a mixture of all spaces' functions, which are weighted by their occurrence counts. In this case the affected observations are set to $\mathbf{o} = (S(o), x)$, where $S(o) \setminus G = 0$.

Additionally, this concept can also be useful in other scenarios than compensating for information loss. For example, external factors may impose a specific model structure whose level of detail can not consistently be met by the source data. In this case, a data point in the originally observed sequence may have two or more equivalents in the user model. In a multi-space HMM this deviation in the level of detail can be mitigated without the aid of any heuristics by using a mixture of the applicable actions' spaces. Here, affected observations take the form of $\mathbf{o} = (S(o), x)$, where $S(o) \subset G$.

To model these situation outlined above, a generalised multi-space HMM has to be used, where the emission probabilities are calculated as weighted mixture of the spaces' individual probability functions. The space weights w_g can easily be calculated by sampling the occurrence counts of the user actions, while other methods may also be applicable. Again considering exponential decay functions to determine the probability of making a multi-space observation, the expression given in Equation 3.2 generalises to

$$b(o) = \mathbb{P}(S(o), x) = \sum_{g\in S(o)} w_g \lambda_g e^{-\lambda_g x} \tag{3.10}$$

where λ_g is the rate of the exponential decay function in space Ω_g. Similarly, when using Gaussian functions with mean μ_g and standard derivation σ_g^2 in space Ω_g Equation 3.3 generalises to

$$b(o) = \mathbb{P}(S(o), x) = \sum_{g \in S(o)} w_g \frac{e^{-(x-\mu_g)^2/2\sigma_g^2}}{\sqrt{2\pi\sigma_g^2}}. \tag{3.11}$$

The final generalisation concerns the observation probability functions, which are in principle not restricted to being one-dimensional. Consequently, in a feature rich environment the definition of observations can be extended further to the form $\mathbf{o} = (S(o), x^m)$, where the continuous part is a high-dimensional vector $x \in \mathbb{R}^m$. When additionally the emission function N_g of space g is set to an m-mixture Gaussian function the probability of observing $\mathbf{o}$ is given as

$$b(o) = \mathbb{P}(S(o), x^m) = \sum_{g \in S(o)} \sum_{m=1}^{M} w_g \phi_m N_g(\mu_m, \Sigma_m) \tag{3.12}$$

where w_g is the weight of space Ω_g and ϕ_m is the weight of the m^{th} Gaussian mixture component with mean vector μ_m and covariance matrix Σ_m.

3.3 Applying the Framework to a Use Case

Based on the framework's general architecture, the three possible modelling variants can easily be utilised by configuring the framework accordingly. However, depending on the use case and research question additional implementations may also be necessary, for example, to parse, store and internally process a particular type of raw data. In the following sections some of the basic design choices when applying the framework in practice are discussed. A concrete realisation of these concepts is afterwards presented in the respective application chapters.

Since hidden Markov models generally do not provide automatic means to perform feature selection, one of the central modelling prerequisites is to conduct this selection manually. The procedure can be based on domain knowledge or other external factors, for example availability—each modelling parameter needs to be tuned using a sufficient amount of training data. In use cases where an objective performance metric is available, different feature sets can also be evaluation based on their effect on the models' performance. Depending on the selected feature set the best suited modelling variant can be chosen, i.e. discrete, continuous or multi-space.

When features from a continuous value range are included in the model it is additionally necessary to determine which probability function is fitting the data optimally. To that end, the framework supports Gaussian functions, Gaussian

mixtures and exponential decay functions in conventional as well as multi-space form, which cover many possible application scenarios. In special cases other functions might be more appropriate, for example, Laplace or logistic functions, which are currently not implemented. However, if necessary these can straightforwardly be added by manipulating the calculation formula of the tuning algorithm (Line 10 in Listing 3.1). If multi-space probability distributions are required for the modelling problem at hand, respective considerations about choosing appropriate probability functions need to be conducted for every space. It is also worth noting that in principle different functions can be chosen for every state and space depending on the data structure.

Apart from the selection of appropriate emission functions, it is also important to tailor the structure of the hidden process towards a specific application. In the most general case a model will permit transitions from every hidden state to every other hidden state including itself at any time step (ergodic model). In other cases external factors or prior knowledge dictate that the hidden process needs to follow a certain structure. If a non-ergodic model is required, it can be generated by exploiting the tuning algorithm's property that any model parameter initialised with zero probability will remain unchanged for the entirety of the procedure. Consequently, specific transitions of the hidden process can purposefully be rendered impossible, allowing to shape the process as needed.

Finally, the fact that the parameter tuning algorithm can get stuck in local optima needs to be considered when creating and exploiting HMMs. Since there generally exits no analytical solution to the problem, this framework provides a rather brute force solution. Specifically, for modelling scenarios where no external factors are available to base initial model configuration on, a *factory class* is provided, which can be used to create an arbitrary number of randomly generated initial models given any predetermined hidden state count. After repeating the tuning procedure for every candidate model, the best performing model can be used for further analysis.

However, an evaluation criterion is essential to conduct this procedure, which needs to be determined with respect to the use case. Sometimes the models may be used for tasks generating quantifiable results, for example classification. Here, model selection criterion comes naturally in terms of classification performance and can directly be exploited. In other use cases analytically comparing models' performance may be necessary, for example, by using the Kolmogorov–Smirnov test to determine the goodness of fit. However, this generally introduces an additional cross-validation, which in turn increases the demand on the size of the dataset and increases computational complexity.

In the next part of this thesis two practical applications of the HMM-based sequence modelling framework are presented. In both cases HMMs constitute an

interesting alternative to other methods commonly used in the respective domain. In the first application search engine log data is used to build quantitative models of user actions in session search. These models can be used to gain insights about the search process augmenting cognitive models generated based on focused user studies. Since there is a large body of prior qualitative studies on the subject, a continuous HMM is chosen as the modelling basis, while the user actions are coded in the model's hidden process. This allows usage of comparably simple action time emission functions, which can be related to prior findings more easily.

The second application focusses on one of the key contemporary problems in social media, the strong prevalence of rumours and potentially false and misleading information. Specifically, HMMs are used as a classifier for rumour detection and veracity determination based on a sequences of tweets. In this case, the presented application is in contrast to the related work in the domain by classifying rumours without making direct use of any text based features. Since there exists no prior knowledge about the temporal patterns in stance distributions, a multi-space HMM setting is used for this task. Furthermore, by comparing the multi-space models to their discrete counterparts, the benefit of the joint modelling of the temporal component can be evaluated directly.

Beginning with the search session models, each application is presented by first introducing the dataset used, followed by a description of the concrete modelling setup. Next, the results and a discussion are given before closing the sections with an outlook on further research ideas in the area.

Arising thereby, the applications demonstrate the general applicability of HMM-based sequence modelling in their domains by generating valuable findings. Furthermore, they act as a demonstration of the HMM framework's flexibility in terms of input data type, application domain and nature of obtained results.

Part II

Applications

CHAPTER 4

Search Phase Identification

In this first practical experiment the temporal ordered search phase approach is followed as it is described in the literature (see Section 1.2.1), using HMMs to quantitatively describe and detect these phases. Automatically identifying search phases in complex session search scenarios in a quantitative manner will fill a research gap that has been evident in the domain since the first qualitative studies investigated on user behaviour in analogue information systems. Past as well as current efforts in understanding users' behaviour foremost focus on small scale studies in controlled environments. Furthermore, the most influential user models in IR focus on cognitive aspects including the users' feelings, thoughts and moods (Kuhlthau, 1991). While insights generated by these qualitative models also play an important role in building efficient and effective search systems, the understanding of users' behaviour can additionally be furthered by quantitative models.

This holistic view on the search process will finally be opening up new opportunities for developing new and better search systems. For instance it could play an important role in an eventual practical application of the interactive probability ranking principle (Fuhr, 2008). Although it has been around for a decade, little advancements have been made towards systems that actually fully develop its concept. However, doing so would potentially be beneficial to end users as well as system designers in various ways. Ideally, a system would be able to determine the current search phase as a categorical representation of the users' mental state in real time during session search. Given the current search phase the system

would then be enabled to adapt itself to the situation, for instance, by providing appropriate tools and adjusting the ranking algorithm. Furthermore, such an adaptive system can provide guidance to the user, for example, by making them aware of common pitfalls in their current search situation or issuing a warning if distraction is imminent.

Naturally, developing such a system is a tall order and out of the scope of this thesis. However, using the general HMM architecture a first step towards automated search phase detection was made by conducting a post-hoc search engine transaction log analysis. Thereby the overall procedure was as follows: First, it was investigated if patterns in the users' actions and their durations can successfully be detected by using two types of HMMs. Second, the user actions are classified into two search phases with respect to their occurrence likelihood and expected durations. Afterwards, the quality of the modelling of user action durations is evaluated using a cross-validation procedure, comparing the models' predictions with the real search engine log data. Finally, this qualitative understanding of session search is related to the previous understanding of the search process as it is derived from the cognitive models. Parts of the work presented here have also been previously published by Dungs and Fuhr (2017).

4.1 Modelling Session Search as a Two-Phase Process

While the cognitive models in the related work generally feature a higher number of search phases (Kuhlthau, 1991; Meho and Tibbo, 2003; Vakkari, 2001), in this experiment only two phases are defined. The contribution of this HMM-based two-phase approach is to move from describing search in terms of abstract concepts and users' feelings to utilising quantitative measurements. Limiting the scope to two phases ensures a manageable model complexity in terms of state and parameter count, which is especially important for tuning the parameters in case training data is limited (Rabiner, 1989). Although the two-phase setup may not accurately map to the search process in reality, it is supposed to still provide a useful basis for further HMM-based session search models built from real world search engine logs.

The general intuition behind the two search phases approach was synthesised from the cognitive IR models, mainly the *information search process* (ISP) introduced by Kuhlthau (1991), which splits a search session into two parts. The first part is viewed as a phase of topic familiarisation by using general queries followed by a rapid inspection of a high number of documents, whereas the second part is characterised by increased user effort with respect to inspection of the documents' actual content.

Consequently, the first search phase titled *Searching* unifies the phases Exploration and Formulation from the ISP by Kuhlthau into a single phase. While the ISP has a strong focus on users' feelings and thoughts during the search process, Kuhlthau also specifies that user actions in the Exploration phase are commonly targeted towards achieving a general familiarisation with the topic. Since the experiment described here use data from a web-based search engine the most common user actions in this phase are expected to be issuing a query and viewing a high quantity of document snippets. Furthermore, the Formulation phase is defined as "the turning point of the ISP" (Kuhlthau, 1991, page 7) after which user action patterns are expected to have changed. This pattern change is exactly what the hidden Markov model is expected to be able to capture successfully.

Additionally, the second search phase titled *Finding* is defined analogue to the Collection phase of the ISP. Following Kuhlthau's definition the Finding phase is characterised by an increase in search effectiveness and efficiency. In this phase the user is also expected to be able to specify the information need in a more focused manner. Related to the web-based searches that are investigated here, the main assumption is that users spend their time studying documents in more detail. Furthermore, users are expected to find relevant documents more quickly as they can formulate better queries after the initial learning phase.

It is also worth pointing out that other quantitative manifestations of established cognitive models are feasible using the general HMM architecture. For instance, an alternative phase definition and the addition of a third phase would already yield a model implementing Vakkari's (2001) view on the search process consisting of the three phases pre-focus, focus formulation and post-focus.

In the following, based on this general two-phase model of session search, the user actions and their respective durations are used to determine the point in time when a user transitioned into the second phase of the search session. Furthermore, by learning HMM parameters based on real search engine transaction log data, certain properties of the phases can be estimated. Specifically, by examining state transition probabilities, estimates about users' next action and the expected time until observing a relevance signal can be made.

4.2 Dataset Description

This experiment is based on transaction log data from the social science digital library *sowiport* which was developed by the *GESIS Leibniz Institute for the Social Sciences*[1] and discontinued in late 2017. The data comprises anonymised log data from a period of 15 month beginning in early 2014 to mid 2015. The

[1] `https://www.gesis.org/en/de/institute/`—last accessed 24.10.2018

sowiport system used Javascript to capture the users' interactions with the web interface, producing in total more than one million individual entries allocated across 32185 sessions. Each entry specifies one of 55 possible user actions and is associated with a unique user ID hash and a timestamp. An example of raw entries is given in Figure 4.1 as a excerpt from the database containing the log.

id	log_id	session_id	action_count	session_length	date	action_length	mapping_action_label
343	4232524	3c02ghShomf9gjvevcqm5vjv34	30	6364	2014-11-13 14:14:01	7	search_advanced
344	4232524	3c02ghShomf9gjvevcqm5vjv34	30	6364	2014-11-13 14:14:01	7	search
345	4232524	3c02ghShomf9gjvevcqm5vjv34	30	6364	2014-11-13 14:14:01	7	search_change_facets
350	4232532	3c02ghShomf9gjvevcqm5vjv34	30	6364	2014-11-13 14:14:08	10	goto_advanced_search
351	4232576	3c02ghShomf9gjvevcqm5vjv34	30	6364	2014-11-13 14:14:48	19	search_advanced
352	4232576	3c02ghShomf9gjvevcqm5vjv34	30	6364	2014-11-13 14:14:48	19	search
356	4232597	3c02ghShomf9gjvevcqm5vjv34	30	6364	2014-11-13 14:15:07	-1310	goto_advanced_search
357	6075011	3c02ghShomf9gjvevcqm5vjv34	30	6364	2014-11-13 13:53:17	0	goto_google_books
504	5379721	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 21:44:22	28	view_record
506	5379733	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 21:44:50	7	view_record
508	5379735	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 21:44:57	15	search
511	5379738	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 21:45:12	7	view_record
513	5379741	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 21:45:19	9	view_record
515	5379746	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 21:45:28	193	view_record
517	5379791	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 21:48:41	4	view_record
519	5379792	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 21:48:45	4	view_record
521	5379795	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 21:48:49	15	view_record
523	5379800	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 21:49:04	698	view_record
525	5379963	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 22:00:42	34	search
528	5379983	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 22:01:16	19	view_record
530	5379988	3c0g57ra99r6jtsrkbc291bmr2	13	5713	2014-12-16 22:01:35	0	view_record
883	8746654	3c1d1ia9v19680k9ha526tst34	21	1813	2015-04-24 12:29:31	8	goto_home
884	8746698	3c1d1ia9v19680k9ha526tst34	21	1813	2015-04-24 12:30:06	13	search
887	8746754	3c1d1ia9v19680k9ha526tst34	21	1813	2015-04-24 12:30:42	10	search
890	8746758	3c1d1ia9v19680k9ha526tst34	21	1813	2015-04-24 12:30:52	3	view_record
892	8746760	3c1d1ia9v19680k9ha526tst34	21	1813	2015-04-24 12:30:55	66	view_record

Figure 4.1: Sample of the unprocessed sowiport log extracted from the database

4.2.1 Preprocessing

Before starting the analysis this particular dataset had to be pre-processed to a usable format by removing the unneeded data columns, only retaining the user actions and their respective durations. These pairs are arranged in sequences in temporal order where each sequence represents an entire search session beginning with the first query.

The original sowiport log is very fine grained distinguishing 55 different user actions in total. This comprehensive action set is difficult to describe in a hidden Markov model since each observation value has to be modelled separately as a hidden state emission when using discrete emission HMMs. This increases the quantity of parameters that are to be learned and therefore the risk of over adaptation during model training. Moreover, a larger quantity of states and emissions increases computational complexity and makes models harder to comprehend analytically.

Additionally, to a general ambition to keep the observation alphabet at a reasonable size, the original log also contains many expendable events. For instance user

action categories are often redundant with respect to the users' actual intent, for example, by differentiating between sending a query by using the author, keyword or institution field. These actions can be unified in a straightforward way without risking information loss with respect to the research questions. Furthermore, the log also contains a number of entries which are not related to the actual search process (e.g. user login, password change, etc.) which were also excluded from the analysis.

The work presented here acts as a proof of concept for HMM-based search phase modelling. As such it is limited in scope, leaving potential granularity refinements for future work. Consequently, a mapping of the 55 sowiport user actions onto four basic prototypical user action categories has been introduced. Specifically, the category *Query* summarises all actions related to formulating and issuing a query (e.g. accessing advanced search features, using facets or filters). *Snippet* and *Abstract* are defined as the categories of actions related to viewing a document's snippet and its abstract, references or citations respectively. Finally, *Mark* comprises all implicit relevance signals, for example, exporting a document or saving it in the personal favourites list. The full list of all action mappings and exclusions is detailed in Appendix A. Although some information is lost due to the mapping, the four basic categories were chosen carefully with respect to the research question and sufficiently cover the user actions present in the cognitive models.

In the final preprocessing measure, all short sequences containing no queries or only few relevance signals were removed from the transaction log, since the aim of this experiment was to model complex session search. After experimenting with different cut-off values, the minimal relevance signal count was set to four, marking the best trade-off between retaining the largest possible portion of the sessions while keeping only sessions that are most likely to feature search phases. Additionally, the data was also cleaned of missing or implausible values (i.e. negative action durations), where each afflicted session was removed from the log. However, a close investigation of the log data revealed that Mark actions are over proportionally often logged without any reasonable duration (i.e. -1). Since these actions are most important for the analysis but at the same time the rarest occurring user action, excluding all affected sequences was not practicable. Instead the duration for the Mark events was entirely removed from the data set and in the following a constant Mark action duration of one second is assumed. In total 1642 cleaned sessions containing 257,592 user interactions met all the requirement discussed above and are used as the data corpus for this experiment.

4.3 Setting Up Modelling Parameters

Since search phase progression in complex search sessions is expected to be one-directional, the models' state transition probabilities need to follow certain constraints to ensure that the models conform to this assumptions. However, random start value initialisation for Baum-Welch parameter optimisation as described in Section 2.2 generally yields *ergodic* models, in which every state is connected to every other state by a direct transition. This violation of the one-directionality assumption of phases can be counteracted by using non-ergodic HMMs.

In a non-ergodic HMM start values are not assigned entirely at random. Instead, certain transition probabilities $a_{ij} \in A$ are deliberately set to zero. Because of the iterative nature of the Baum-Welch algorithm, specific expressions in the algorithm will always yield zero when a parameter is initially set to zero, meaning that a transition probability $a_{ij} = 0$ will remain unchanged during the entire parameter optimisation process. This property of the algorithm can be exploited to define models of a particular shape best suited for the task. The same principle can also be used to specify which emissions are observable in each hidden state of the model. Popular examples of non-ergodic models are so called left-to-right or Bakis models used in speech recognition (Bakis, 1976). HMMs following the Bakis constraint have the property that as time increases the hidden state index never decreases.

For search phase modelling a semi-Bakis model is used, where the hidden process consists of ergodic sub-graphs which are combined following the Bakis model structure. Figure 4.2 visualises the semi-Bakis flow of the hidden process in terms of the model's state transition matrix A, where all transition probabilities $a_{ij} > 0$ are indicated by dark blue coloured squares while probabilities $a_{ij} = 0$ are depicted in light blue. In the example each of the three phases consists of three hidden states. However, in principle phase count is not fixed and phases can have different quantities of hidden states. Depending on the actual model structure the states' emission probabilities are initialised carefully so that the state-emission pairs accurately reflect the expected work flow in session search.

4.4 Discrete Emission HMM

Overall two variants of semi-Bakis hidden Markov models were created. Both models consist of two ergodic sub-graphs representing the search phases. The sub-graphs are connected by a single one-directional state transition. Although in principle incremental phase transitions could occur from any state in a sub-graph without violating the modelling assumption, for the scope of this experiment

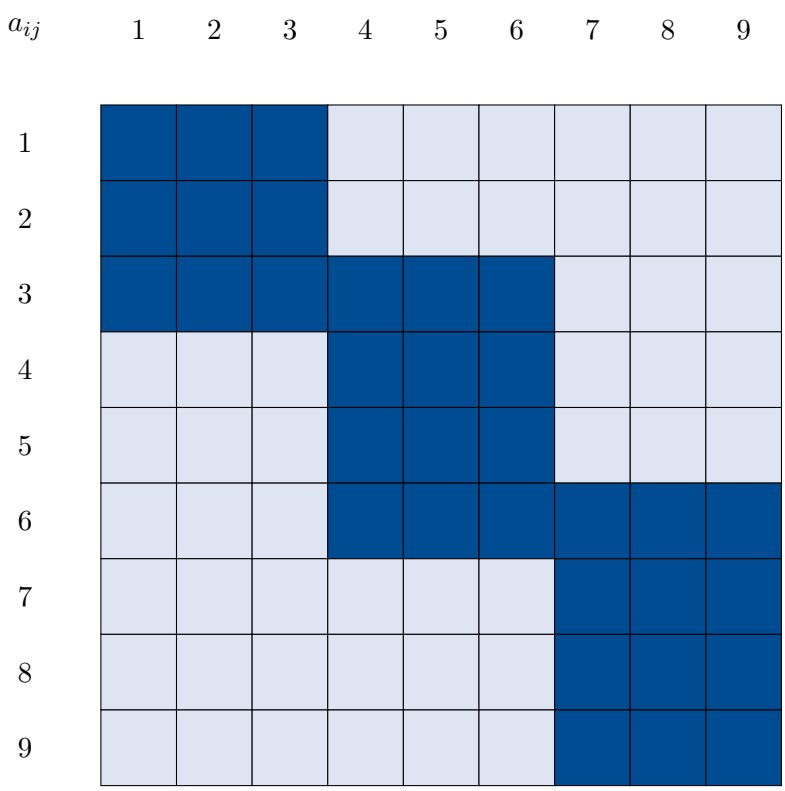


Figure 4.2: Visualisation of a semi-Bakis model's state transition matrix A

usage of a single transition point was preferred. The main reasoning is that this makes the phase transition more explicit and sub-graph transition probabilities can more easily be compared between phases.

4.4.1 Model Description

In the discrete variant session search is modelled based on the observed user actions alone. Consequently, in this setting the model's observation alphabet is defined as $E = \{query, snippet, abstract, mark\}$. Furthermore, this model has four hidden states, two for each search phase. The states' phase membership is indicated using the indices s and f for the phases Searching and Finding respectively. Additionally, according to their designated role in the search model, the states are named $Work$ and $Mark$. The $Work$ states are used as an assembly of the three observable user actions not associated with relevance signals (query, snippet, abstract). This states' emission probabilities are learnt during parameter the optimisation process with the exception that the probability of observing

a mark action is set to zero. Consequently, $Mark$ states are used to explicitly model the relevance signals, i.e. their emission probability for Mark user action is set to 1.0. Conforming to the stochastic emission matrix property, all other emission probabilities are set to 0 in both Mark states. The transition probability matrix is initialised with the respective zero values to create a semi-Bakis search phase model. Additionally, it is worth pointing out that due to the explicit modelling of Mark actions the respective states in the final graph do not feature a looping transition, since the log data did not contain any consecutive marks.

4.4.2 Results

Before discussing the discrete models in detail, it is important to recall that HMMs are probabilistic and built based on the training data by using an EM algorithm suffering from the local optimum problem. As a result, the discussed HMMs should not be considered as the absolute solution to the problem at hand. However, all major findings presented below also hold when re-training the models using different initial parameters, although these models will inevitably feature slightly different parameter sets. Therefore, focusing on the model's general purport, results are rounded to percentage point accuracy in the discussion below.

Figure 4.3 details the final model after parameter tuning which categorises search actions using the two states Work and Mark for each search phase. For the remainder of this thesis, models' hidden states are depicted using blue circles and emissions are rendered as boxes in light yellow. If a model has a well-defined start state, this state is marked using concentric lines. Investigating the difference between both Work states in this model, two major observations can be made.

On the one hand, by looking at the transitions probabilities, it can be seen that with a probability of 8% a transition to Mark state is more likely to occur in state $Work_f$ compared to the 3% occurrence probability in $Work_s$. Therefore, on average users require fewer actions in Finding phase to reach the next relevant document. On the other hand, comparing the emission probabilities between both Work states, it can be seen that on average users perform fewer Query actions (20% vs. 17% occurrence probability) as well as Snippet actions (72% vs. 70%) in Finding phase. Furthermore, in $Work_f$ the model predicts a 5% higher probability of performing an Abstract action (13% vs. 8% in $Work_s$). Furthermore, in this model the user is transitioning to the second phase with a 10% probability after making a Mark action.

To further the quantitative understanding of session search, in the next section a more elaborate model based on continuous emissions is introduced.

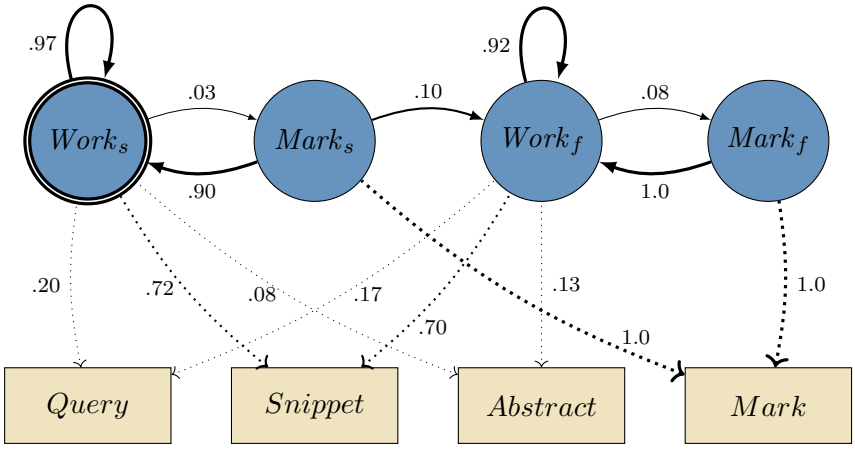


Figure 4.3: The search phase model composed of four states

4.5 Continuous Emission HMM

Discrete emission HMMs have the advantage of a simple layout which is also easy to comprehend visually when the hidden state count is low. Naturally, easiness of use is offset by a limited model expressiveness since continuous features can not be included. As the search engine transaction log also contains information on a continuous scale in form of the user actions' durations, an alternative modelling configuration using a continuous emission set was created following the definition given in Section 2.2.1. While the discrete emission HMM puts an emphasis on the changes in state transition and emission probabilities between the search phases, the continuous model supplements the former by also being able to model changes in action durations between the two search phases. Capturing these changes is also an important factor when considering the overall potential applications of successfully implementing quantitative session search models. Many of these applications have time and efficiency related aspects, for example, predicting the time to task completion, which could be an essential part of an extensive user guidance system.

It is also important to point out that apart from continuous emission HMMs, there are also other variants of HMMs that could have been utilised for the task at hand. Continuous *time* HMMs on the one hand extend the expressiveness of the models used so far by generalising the progression of the hidden process from discrete time steps to a continuous process. In principle a continuous time HMM could be created where each state in the model captures the user actions' durations. However, these models are fundamentally different in nature and require an alternative set of algorithms for solving the HMM-related problems which are

not further explored in this thesis. On the other hand, multi-space HMMs are capable of unifying discrete and continuous emissions on a more general level. However, these models also require a considerable larger parameter set. This is not only introducing a higher computational complexity when performing model training, but also increases the risk of overfitting when state-space combination parameters are trained using few data samples. Therefore, out of the discussed principal modelling techniques, continuous emission HMMs are best suited for the task.

4.5.1 Model Description

The second model's hidden process in total features eight hidden states, i.e. one state for every $Action \times Phase$ combination, to learn the effect of current search phase on the user actions' durations. For easier reference hidden states are named following the scheme $Action_{phase}$, e.g. state Q_s for all Query user actions in phase Searching or state M_f for relevance signals that are observed in Finding phase. The model is arranged so that each search phase is modelled by a four state ergodic sub-graph consisting of the applicable user actions. The transition between phases is one-directional and solely possible from state M_s, ensuring the availability of an explicit phase transition point in the final analysis. A graph of this model's hidden process is depicted in Figure 4.4. Same as it was the case in the discrete model, the Mark states in this model do not contain a loop due to the fact that two relevance signals never occur consecutively in the data.

For modelling user actions' times as a continuous feature this model's states use Gaussian probability distributions (Gaussian PDFs) as emission functions. For each Gaussian PDF used in the model a mean and a variance parameter has to be initialised. For the actions Query, Snippet and Abstract these parameters have been set uniformly for both phases and are respectively based on the individual mean and variance values calculated using the entire dataset. Since in the transaction log the Mark actions' times are very noisy, these are excluded from further consideration and a constant action duration of one second is assumed in both phases. During Baum-Welch parameter optimisation the emission functions are tuned alongside state transition probabilities to best fit the data. Therefore, the final output includes search phase specific mean and variance parameters for each probability density function.

It is worth pointing out that although using a Gaussian PDFs for modelling times strictly speaking violates the requirement given in Equation 2.9—negative times are impossible—in practice this has little effect on the outcome. Nevertheless, an additional model has been created, replacing the Gaussian PDFs by a decay function which results in a mathematically sound model. However, comparing the resulting models after parameter optimisation, no significant difference could

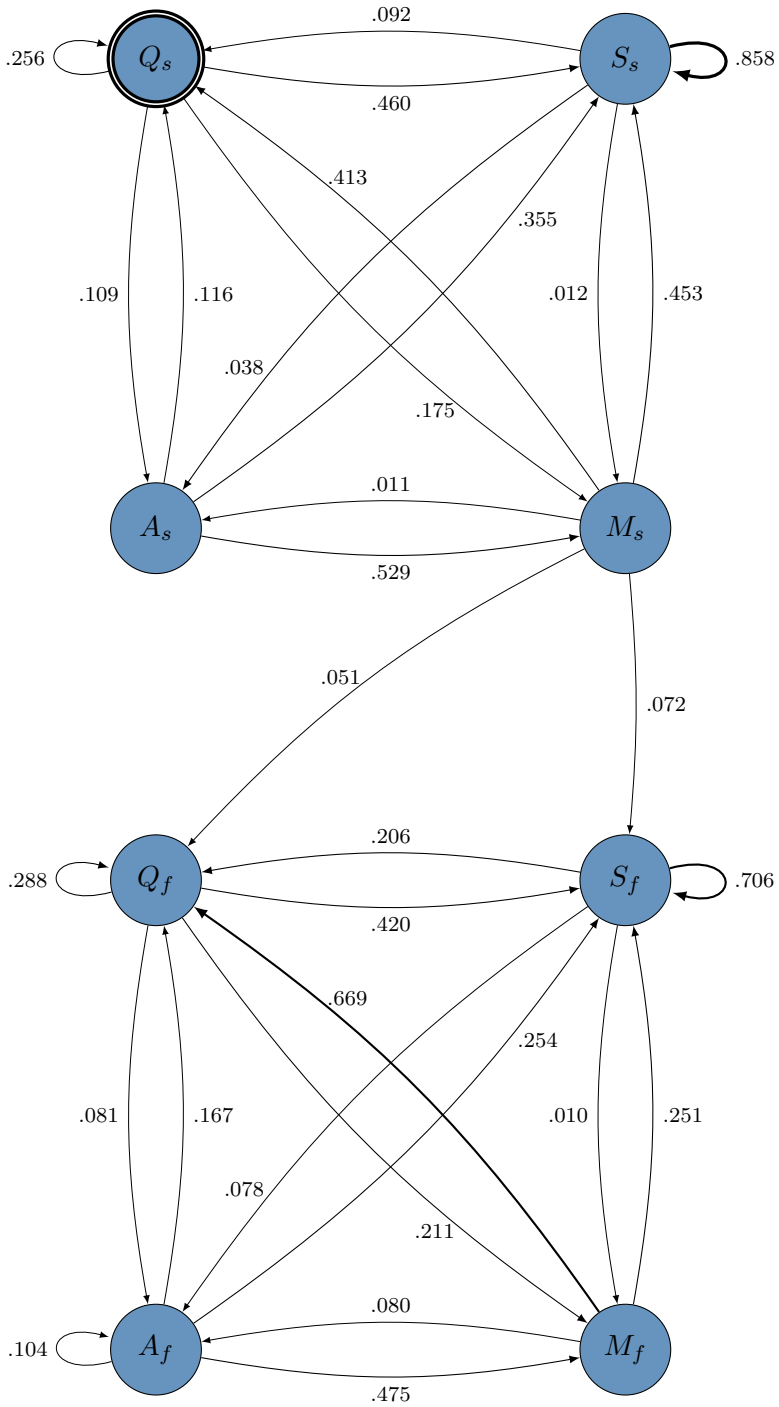


Figure 4.4: Hidden process of the search phase model composed of eight states

be observed. In the following, results are discussed based on the Gaussian PDFs because of its comparably more intuitive representation by mean and variance values.

4.5.2 Results

The continuous HMM's state transition probabilities are shown in Figure 4.4. Since this model uses Gaussian PDFs as the states' emissions instead of an emission probability vector, these functions are omitted in the figure and instead presented in terms of their mean and variance values in Table 4.1. Same as for the discrete model, the states' transition probabilities differ with respect to the search phases. Comparing the transitions leading to states Q_s and Q_f it can be seen that users are more likely to reformulate their queries in phase Finding since all incoming edges of state Q_f have higher probability. Interestingly, this also includes the probability of looping in this state. Alongside this observation, the model features a reduced probability of observing a Snippet action in the Finding phase since all incoming edges of S_f have a lower probability compared to the edges reaching state S_s. Most strikingly is the difference in looping behaviour concerning Snippet actions with and probability decrease from 86% in phase Searching to 71% in Finding phase. However, while viewing on average fewer snippets, users identify more potentially relevant documents which is indicated by an increased probability of transition $S \rightarrow A$ in phase Finding (8% vs. 4% in Searching phase).

However, action A_f actually has a lower probability of resulting in Mark compared to A_s (48% vs. 53%). Furthermore, viewing a snippet following an Abstract action is less likely in Finding (25% vs. 36% in Searching). Instead, in Finding the model predicts a 10% chance of looping Abstract actions—which is very improbable in Searching—as well as 17% probability of transitioning to Query (up from 12% in Searching). The most likely user action after finding a relevant document (state M_x) also changes between phases. When neglecting phase transition probabilities, in Searching the user is most likely to continue using the current query by viewing more snippets (45% probability) followed by reformulating the query with a likelihood of 41%. In Finding phase the most dominant follow-up action is Query with a 67% probability, followed by Snippet with a 25% probability. However, the actual search phase changes, represented by the transitions from M_S to any Finding state X_f, have to be considered as well. Here it can be seen that users are more likely to continue viewing snippets after transitioning to Finding phase than issuing a new query ($M_s \rightarrow S_f : 7\%$ vs. $M_s \rightarrow Q_f : 5\%$).

After presenting the model's transition probability matrix, in the next paragraph the emission functions are detailed. With the exception of Mark actions the transaction log contains all actions' durations which are modelled by search phase

Table 4.1: Mean and variance of user action durations in both phases given in seconds.

Search phase	Query		Snippet		Abstract	
	Mean	Variance	Mean	Variance	Mean	Variance
Searching	7.5	25.4	2.5	2.7	36	218.9
Finding	2.2	3.3	1.6	1.7	23	266.2

specific Gaussian PDFs. Each Gaussian function can be fully described given two parameters, mean and variance. Table 4.1 details the final values for each trained PDF in both search phases.

Overall, it can be seen that user actions are on average much shorter in Finding phase. Especially Query actions are performed more quickly in Finding using less than a third of the time compared to Searching (2.2 vs. 7.5 seconds). Furthermore, average time of Abstract actions is cut by 50% and time spend for Snippet actions by 36% in Finding phase. Looking at the functions' variances large differences can be noted both between user actions as well as between search phases. Snippet actions show the smallest variance across all observations. When considering search phases, it can be observed that additional to the overall quicker execution times of Snippet in Finding, actions taking much longer than the average also occur less often (lower variance). Query actions' times also show a smaller variance in Finding. However, in this case the difference is much more distinct (25.4 Searching vs. 3.3 Finding), which means very long Query durations are far less likely to happen at later stages of the session. Considering Abstract actions' times, first of all the very high variances have to be noted, which exceeding the other states' by one to two orders of magnitude. Furthermore, contrasting the observations made earlier, the Abstract action's variance increases in Finding phase by almost 22%.

4.6 Discussion

In the following section the outcome of the experiment is discussed with respect to the initially formulated assumptions of two-phase session search by investigating effectiveness and efficiency of the users' actions. Afterwards, the models are used to analyse the phase transition point of search sessions as well as to estimate expected times of future actions in a search session. This chapter is continued with a discussion of the limitations of HMM-based search session modelling in general and with respect to this particular dataset before concluding by pointing out additional research questions for potential further experiments.

4.6.1 Search Effectiveness in Finding Phase

As motivated in this chapter's introduction, the main reasoning for modelling search as a two-phase process was based on the general idea that user initially need to familiarise themselves with the topic in question when being confronted with complex information needs. This first phase is then followed by a second phase in which users find documents with increased efficiency and effectiveness. Overall, both presented models of two-phase session search support these assumptions.

Already the very simple discrete model is able to show one of the major differences between the search phases: The increased probability of Mark observations in the second phase of the session, which indicates that users are requiring on average fewer actions to reach the next relevant document. This finding is in line with the related work. For example Spink et al. (1998) examined different dimensions of users' document relevance judgements and found that perceived relevance is also influenced by previous interactions in the search process. Furthermore, Pharo and Nordlie (2012) performed a study in which a set of fixed book search tasks is assigned to the participants. One of the findings was that searchers with little topic knowledge had to inspect significantly more books before finding a relevant item compared to more experienced searchers.

Additional to increased probability of observing a Mark action, the discrete model also shows additional changes in user behaviour with respect to the search phases, which can be determined by considering the actions' occurrence probabilities in the respective Work states. In particular, users issue fewer queries and view fewer snippets in Finding phase. Instead, users more often investigate documents' abstracts before making the final relevance judgement in the second phase. This indicates that in the second search phase users are more satisfied with the result sets their queries generate, since fewer documents can be confirmed as irrelevant by solely viewing the document's snippet.

Thoroughly interpreting the final continuous model is not as straightforward as analysing the discrete model since several additional aspects have to be considered. Firstly, because of the introduction of additional hidden states the transition probability matrix is more complex, making it harder to get an analytic understanding of the likelihood of specific state sequences. Additionally, by introducing the time component, this model reaches a superior expressiveness over the discrete version which also has to be investigated with respect to the two-phase session search assumption.

Despite the fundamentally different feature set used in this experiment, overall the observations made using the discrete model hold. Moreover, when analysing the extended model in terms of transition and emission probabilities it also becomes evident that users search more effectively in the Finding phase by formulating bet-

ter queries while also acting more efficiently. Increased effectiveness is expressed in the apparent increase in likelihood of transition $S \rightarrow A$ in Finding phase. This is indicative of result sets that generally feature more documents worth of extended investigation by viewing their abstracts. It is worth noting that both models have slightly deviating probabilities for entering the second phase—10.3% in discrete model vs. 12.3% summed up over the continuous states. However, the continuous model is not merely an extension of the discrete model. Rather it is of entirely different nature by including the times. Additionally, slight parameter deviation are expected in probabilistic models and do not devalue the results.

4.6.2 Search Efficiency in Finding Phase

The other central result is that efficiency of search also increases in phase two because the expected mean time to find the next relevant document is lower in Finding phase. This can be confirmed by comparing probability distribution functions between the search phases which shows on average overall shorter durations for all actions. In absolute terms most time is saved when actions related to the document abstracts are performed. Assuming that most final positive relevance judgements are made based on the documents' abstracts, reduced duration of abstract actions can also be an indication of an overall progression of users' mental representation of the search task or their information need respectively. If during the course of a search session users' understanding of the search task deepens they supposedly also become able to more quickly deem a document as relevant for this specific task—or to dismiss it from further consideration for that matter. When also considering the expected times for Snippet actions in both search phases, the proposed explanation of the observed user behaviour becomes even more conclusive.

In relative terms the largest efficiency improvement can be observed regarding Query actions. Much shorter query formulation times back up the assumptions that the later stage of session search mainly features small incremental changes of the submitted queries, whereas in the initial phase users have to invest increased mental effort into specifying queries suitable for satisfying their information needs.

4.6.3 Considering Variance in Duration of Actions

Certainly, to complete this models' analysis the action duration probability functions' variances have to considered as well. Generally regarding that the duration of an action can not have a negative value, it becomes apparent that a higher variance can only connote an increased likelihood of *longer* action durations.

When examining variance given in Table 4.1 it can be noted that Query action

duration variance is considerably higher in Searching phase. Again following the assumption of a topic familiarisation phase larger variance could occur because users need to think more about the task in the beginning of a search session. Elevated query durations might also be a quantitative representation of the anomalous state of knowledge (ASK) hypothesis formulated by Belkin (1980). According to the ASK, users enter a search session due to of a perceived problem regarding any part of their knowledge structure (e.g. lack of or inconsistency within the information) while simultaneously being unable to specific the problem precisely at first. Therefore, the first few query may produce only unsatisfactory result sets, prompting users to further increase reasoning effort while formulating the next query. On the contrary, in Finding phase overall less mental effort is required to formulate queries, which is indicated by the low variance value meaning very time intensive query actions rarely happen.

In total, document snippets seem to get less attention in Finding as it is also indicated by the lower mean values of the PDFs. Again, while in Searching amplified reasoning may also be triggered solely by viewing document snippets, this phenomenon is less likely to occur after users enter the Finding phase. However, it has to be noted that the Finding phase generally contains less Snippet observations because of the sequential progression of phases. Additionally, as discussed further in the next section, some sessions do not enter Finding phase at all and, therefore, do not provide useful information for parameter estimation. As a result fewer data points are used for tuning function parameters of Finding phase, naturally reducing the variance as well.

However, contradictory to the other actions, variance in Abstract duration is higher in Finding phase—as well as it overall is the highest in the model. One possible explanation for the generally very high variance could be found in the nature of the Abstract action, which requires users to read extended amounts of text. Naturally, a high variance is expected when comparing actions' durations across users because of differences in individual reading performance. Additionally, document abstracts are of varying length, for example, depending on publication type or authors preference which will also inevitably cause situations where making a relevance judgement based on a full abstract becomes substantially more time consuming.

Another possible factor in the elevated variances in this particular case could be that the original transaction log contains a number of noise artefacts. This is especially conceivable when considering the fact that real-world search session data obtained in an uncontrolled environment are used in this experiment. One of the potential scenarios causing unwanted measurements could be as follows:

At first a user performs a (successful) search and then abandons the interface, e.g. because the user was able to satisfy the information need or got distracted by

another task. Keeping in mind that sowiport is a web-based search engine and considering the ability of all modern browsers to support multiple active tabs simultaneously, abandoning does not imply that the session is closed explicitly (e.g. by logging out) or implicitly (e.g. by closing the respective browser tab). Instead, a user may later reuse the already existing tab to continue with the same or a different search. If the session is not automatically terminated by sowiport or if the user does not use an account, any additional actions will be attributed to the prior search session. Therefore, even performing a short secondary task could create log entries with a substantial time offset with respect to the previously conducted action. Having only a few of such misattributed Abstract events could easily have a substantial effect on the model's final parameters. It could additionally be argued that such a scenario is more likely to happen towards the actual end of a search session, which could be an explanation for the unique finding of elevated variance in Finding phase. In future work, data preprocessing should also include a filter to remove any outlier events from the data, e.g. by introducing an action duration cap based on the mean or median value.

It is also worth pointing out that while being large in absolute terms Abstract duration's variance difference is the lowest when considering relative changes between phases. This could indicate that from all three actions considered in the model, the current search phase has lowest impact on Abstract action's durations.

4.7 Pinpointing Phase Transitions

After discussing the final model parameters, in this section an additional interesting problem is addressed. Given that the major assumption in this experiment is the existence of search phases in complex search sessions, it is also worth investigating the actual point in time of phase transition on a per sequence bases. By using the procedure introduced in Section 2.2.2 the Viterbi algorithm can be utilised for this task. The algorithm hereby returns the most likely explanation of the observations as a list of state indices $S = (s_1, \ldots, s_t)$, $s_i \in [0; 7]$ where t is the length of the observation sequence. The first occurrence of a state index $s_i \geq 4, s_i \in S$ marks the phase transition point of a sequence. Any sequence S where $\nexists s_i \geq 4$ is considered to not have transitioned to Finding phase at all.

By using this procedure to calculate phase transition points for all sequences in total 602 or 37% out of the 1642 sequences are determined to reach Finding phase while the other sequences remain in Searching phase for their entire duration. At first glance this high number of supposedly non-transitioning sequences may seem contradictory to the general assumptions of search phase based modelling. However, the assumptions are expected to hold only in case of complex search sessions. In contrast, the conducted experiments are based on an unlabelled

transaction log sample missing any information about the actual user intend or information need. It is reasonable to assume that the log contains sessions that are very heterogeneous in terms of task type and session complexity. However, in this uncontrolled environment only minimal post-hoc effort could be made to exclude trivial sessions before building the models, i.e. by setting the minimal Mark action count to four. Apparently, not all simple information needs were excluded by this procedure. Despite the fact, the models predict two distinct search phases in terms of user effectiveness and efficiency. Furthermore, the continuous model appears to be tuned to successfully handle heterogeneous session types as well since it is not 'blindly' transitioning to the second phase based on any kind of input sequence.

It is also interesting to investigate further on the timing of the sequences' actual transition points. After determining the transitioning point index i the time spend in phase Searching can be calculated by summing up the actions' times for the subsequence S_s with state indices $s_0, \ldots, s_i$. This absolute time spend in phase Searching is denoted as T_{abs}. To exclude the effect of session length additionally the relative transitioning point T_{rel} is investigated where $T_{rel} = T_{abs}/sessionlength$.

Figure 4.5 details the distribution of T_{rel} (y-axis) across all transitioning sequences as a function of session length (x-axis). Clearly it can be seen that T_{rel} is not a function of session length since the whole range of T_{rel} can be observed across the entirety of session length values. Therefore, the continuous model is not only able to cope with heterogeneous session types but also with a wide range of session lengths.

4.8 Parameter Estimation for Interactive PRP

As motivated already above, one of the central intended purposes of automated search phase detection is an advancement towards the eventual estimation of the cost and probability parameters—especially the search time—for the interactive probability ranking principle (interactive PRP) as proposed by Fuhr (2008). While the original model does not include the estimation of the parameters directly, Fuhr identifies three requirements that need to be fulfilled in order to apply the interactive PRP: (1) The complete interaction process should be considered in the model, (2) activities need to be modelled using different costs and benefits and (3) the information needs' dynamics needs to be considered.

The continuous HMM presented here is at least in partial fulfilment with these requirements, since it covers the interaction process from issuing a query until discovery of relevant documents, although other aspects of searching behaviour—

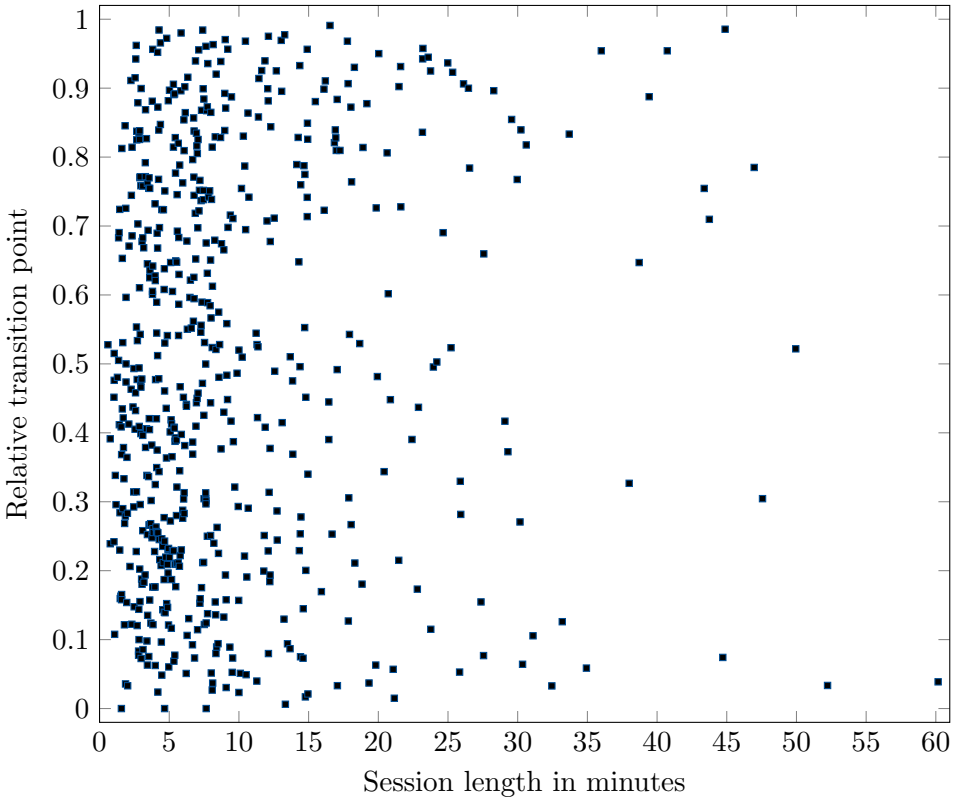


Figure 4.5: Phase transition point in proportion to session length

for example exploitation of relevant items—are not considered, yet. Nevertheless, those parts of the search process that are modelled in the HMM are described in terms of their cost and benefits. For this purpose, an actions' cost can directly be modelled as the actions' duration. The benefit of an action can be calculated indirectly, for example, using the predicted time to the next relevant document, which is influenced by execution of an action. If execution of this action leads to time reduction, this choice has a positive benefit, while the magnitude of the benefit can be modelled by the amount of time that is reduced. The expected time to the next relevant document can be calculated using the HMM by summing all durations of the steps that are necessary to reach any of the models' Mark states.

Fuhr states that the dynamics of the information need should also be addressed in a model, since it can be changed by any positive information. While this is not implemented in the HMM directly, the crucial role of positive information is still emphasised by the fact that in the HMM the phase transitions are linked to observing a positive relevance signal as well. Furthermore, same as a shift in information need influences the model's parameters in the interactive PRP, so

does a phase change in the hidden Markov models of session search.

Nevertheless, the modelling scope needs to be extended to fully cover every aspect of the interactive PRP and a holistic parameter deduction is out of the scope of this work. However, even though the models presented here simplify the search process, they still constitute one of the few instantiations of quantitative models principally capable of estimating the central parameters of the interactive PRP overall.

In the following, parameter estimation using the HMM is illustrated by predicting the time needed to reach the next relevant document, similarly as proposed by Tran and Fuhr (2013). Specifically, sequences s are modelled beginning with the first snippet action after a query or mark action until observation of the next Mark. Using the continuous HMM of two-phased session search, the expected durations of s are predicted for every Mark and evaluated using 10-fold cross-validation on the transaction log. After estimating the HMM parameters on the training data, the actual times taken from the remaining search log are compared to the predictions generated by the HMM. In total four different methods for determining the users' current search phase and estimating the search times accordingly are compared.

Binary Search Phase Decision

To estimate the expected time for each sequence s it is first necessary to calculate the current search phase the user is in, since action times are phase dependent. Note that since s is defined to end when a Mark is observed, the entire sequence either takes place in phase Searching or in phase Finding. The Viterbi algorithm is a standard mean to calculate the most likely hidden state path given a sequence of observations and is also used here as one of the methods to estimate the phase dependent search times. Specifically, based on the sequence of observed categorical user actions, the algorithm returns the most likely hidden state for every action—and thus implicitly also the search phase. The search phase specific emission functions are then used to calculate the expected time of the individual action. Finally, total expected search time is calculated by summing estimated action durations over the entire length of s.

Searching Phase Only

Since the main subject of this experiment is the estimation of times in dependence of the two search phase, it makes sense to investigate on a potential bias of the model. Therefore, in the second approach the user is assumed to remain in Searching phase for the entire duration of the session. This is equivalent to using

only the upper half of the model depicted in Figure 4.4, although, to compensate for the removal of the model's lower half, the state transitions have to be adjusted minimally to maintain a stochastic matrix. Action times are once more generated using the respective states' emission functions.

Finding Phase Only

This third method is analogue to the previously discussed idea, however, in this case the user is expected to remain in Finding phase for the entirety of the search session. A hidden state transition matrix representing this search process can be extracted directly from the main model depicted in Figure 4.4 by using only its lower half. As before, action times are generated using the emission functions.

Continuous Search Phase Decision

The fourth method differs in one crucial aspect from the previously discussed. While so far a binary prediction of the user's current search phase as made for the entire sequence s, here the current search phase is estimated on a continuous scale. To that end, phase prediction is performed individually for every action in s. Afterwards, the ratio of the individual phase predictions is used as a probabilistic estimate of the users' current search phase. The final estimate for the expected duration of sequence s is subsequently calculated as the weighted mean of both phases' emission values.

Comparing the Time Estimates

Figure 4.6 details a box plot of the relative error distributions when the four estimates of expected time to the next relevant document are compared to the actual search times. The respective box boundaries indicate the upper and lower quartile, while the box height is the interquartile range (IQR). Upper and lower whisker markers represent the highest and lowest data point still within the range of $1.5 \times$ IQR of the upper and lower quartile. Horizontal lines in the boxes denote the median error value of the respective method.

When comparing the graphs it can be seen that the times generated using a binary search phase decision are most similar to the Searching phase only method, with a median relative prediction error of 0.09 for the former and 0.13 for the latter method. This means that on average both methods produce time predictions that are too high, while also the methods' overall error distribution has a similar pattern as indicated by the upper and lower quartiles. The similar time

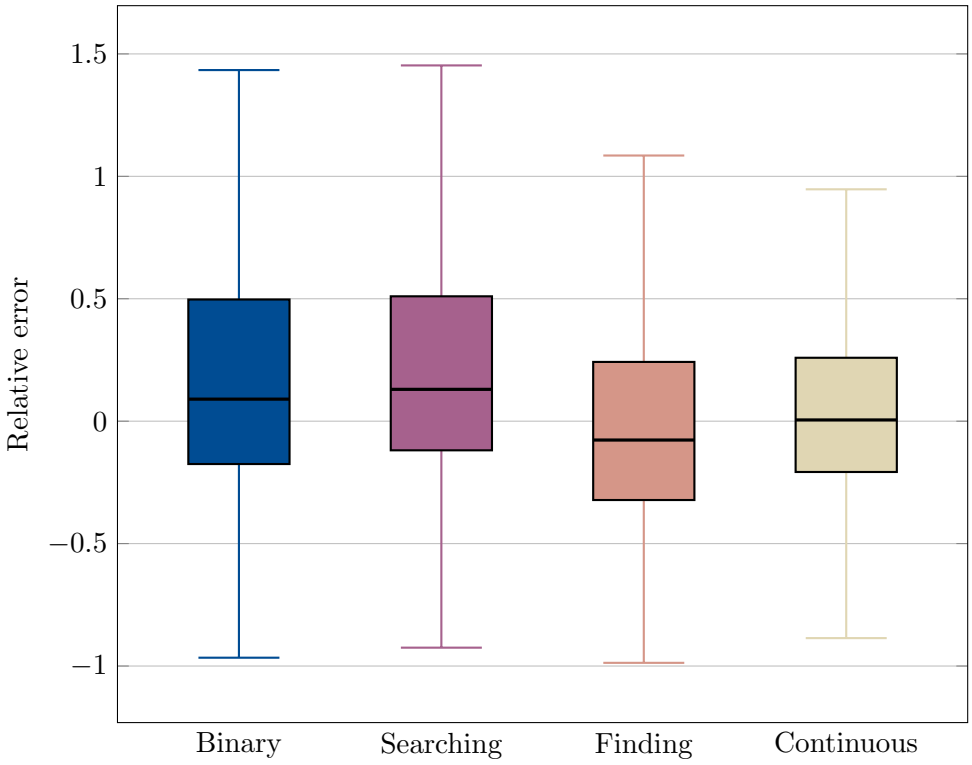


Figure 4.6: Relative error when predicting time to next relevant document

predictions of methods *Binary* and *Searching* also confirm the insight discussed in the previous section: Many search sessions do not transition to the second phase. For non-transitioning sessions both time prediction methods are identical, which explains the overall similarity in the results.

Contrasted to both methods' overestimation of times is the *Finding* method, which has a median relative error of -0.08, yielding on average time predictions that are to low. Furthermore, this method's upper variance is considerably lower compared to methods Binary and Searching. Finally, it can be seen that the continuous search phase decision produces the best results out of the four discussed methods. With a median error <0.01 this method does not only produce unbiased estimates, it is also showing the smallest overall variance in prediction quality.

Although these results can only be regarded as a first step towards an eventual estimation of the parameters of the interactive PRP, it can be seen how a hidden Markov model of search phases could in principle be used for this task. In this case, the continuous modelling of the users' current search phase performs best,

also significantly outperforming the binary phase model in terms of mean absolute relative error across all test runs (385 vs. 561, t-test $p < 0.01$). However, these figures can only act as an approximation of how useful the model can be to estimate the crucial parameters in a setting that is close to the source data. How these findings can be generalised to other systems and task types while additionally also including further search phases is a subject of future work. Other limitations of the models on a more general level are discussed below.

4.9 Limitations

Both models presented in this chapter are able to support a simplified view of cognitive IR models of session search in a quantitative manner by specifying two distinct search phases. However, since these models are only a first step towards a more holistic model including every aspect of session search, they naturally have a number of limitations. In this section these limitation will be discussed while also providing their potential solutions.

The foremost consideration about the models is targeted at the phase transition design choice since confining the transition point solely to implicit relevance signals is potentially a significant oversimplification of the search process. In their current state the models require the user to find at least one relevant document before any actions can be attributed to the second search phase, although this requirement is never explicitly formulated in the cognitive models underlying this experiment. Moreover, if the session does not transition at the first Mark action, a second (and possibly additional) Mark action is required for the phase transition. This may not precisely correspond to users' cognitive processes in reality. An improved version of the quantitative model should be able to reassign the search phase based on any user action, for example, by allowing phase transitions from states Q_s, S_s and A_s.

Despite the promising results obtained with the models, it is also worth pointing out that session search is very likely to not conform to the Markov property of statistical independence of observations. Therefore, following a strict interpretation of the mathematical principles HMMs are technically not applicable for this type of data. Specifically, in the models' current form the phase transition point is determined only based on the current observation value—a single user action and its duration. Since existence or type of any preceding observations are not regarded when making the phase transition, this will inevitably cause inaccuracies in predicted phase transition points, either by too early or too late deeming a session to move to Finding phase. Fundamentally, this limitation can be overcome by using n-order HMMs to assure phase transition decisions are based on the last n observed user actions. However, since higher-order HMMs

also require much larger amounts of training data for parameter tuning which is difficult to obtain, exploring these kinds of models in context of session search is left for future work.

Nevertheless, even in their current form the models successfully capture differences in terms of effectiveness and efficiency between the search phases. One possible explanation for this is that in reality the actual cognitive process undergoes a gradual shift in terms of action prevalence and actions' durations. Following this hypothesis, the presented two-phase models would describe a discretisation of this steady process by capturing both extreme ends of the gradual scale. Therefore, by introducing extended models featuring additional phases, quantitative models can be improved to provide a higher resolution to more closely match the actual continuous state change. Following the semi-Bakis model structure introduced in Section 4.3, the extension to n-phase models is principally straightforward.

Currently, all users and the entirety of all conducted search sessions are used as a data basis to create a unitary model of the search process. As such, the presented models are oblivious to user and task specific differences in the interaction patterns. In their work on predicting search times, Tran and Fuhr (2018) and Tran et al. (2017) have shown how simple Markov models can be personalised to individual users, significantly improving models' performance in the task. Similarly to their approach, user and task characteristics could be accounted for while modelling search phase transitions as well, for example, by calculating user-aware state transition probabilities as well as personalised action durations. However, the implementation of personalised HMMs requires substantial amounts of training data for every user and is therefore out of the scope of this thesis.

Additional to above points, the mapping of sowiport log entries to user action categories has to be addressed, which was introduced to reduce the modelling complexity. Consequently, four categorical user actions were used to describe the users' behaviour in session search while the original log contained 55 different log entries. Even after removing log entries unrelated to the actual search the original logs' vocabulary had a size of 29 items. Therefore, in a follow up study the expressiveness of the models could be enhanced by using an enriched mapping comprised of additional action categories. However, the basic trade-off between model complexity and expressiveness still has to be considered as well. Therefore, suitable feature selection techniques have to be developed first to ensure only non-redundant log entries are retained for further analysis.

Lastly, it must also be pointed out that the models assume constant durations for the Mark action. While this is justified by high levels of noise in the original log files, it still poses a limitation of the models, potentially missing important search phase related changes in user behaviour regarding Mark actions. In future work, this issue could simply be addressed by using a different dataset.

4.10 Conclusion

In this section complex session search was quantitatively modelled as a two-phase process by using hidden Markov models. Overall, two models were presented, both based on a transaction log of 1642 search sessions conducted with sowiport, a social science academic search engine. The first model describes the work flow in session search based on the observed user actions. The main outcome is that users are working more efficiently in the second search phase Finding since they are more likely to find relevant documents while requiring fewer actions.

The more elaborate continuous HMM also includes the user actions' times into the model. Apart from confirming the first models' basic predictions of action probability distribution, the second model also gives assertions about changes in efficiency in the later stage of a search. Overall, the user actions' times in the second phase are expected to be lower compared to the first phase which is in line with established cognitive IR models. Further analysing the distribution of observed times revealed that in the Finding phase extraordinary long action duration are less likely to occur in general. However, an unusual concentration of outlier actions was also described with respect to Abstract actions in Finding phase, which is possibly caused by an artefact in the logging technique.

In Finding phase users are also expected to more often issue a new query compared to earlier stages of the search. This observation can be regarded as quantitative evidence of Vakkari's (2001) idea of a post-focus phase in session search. The author assumes that in this phase users go through a process of rechecking newly acquired information. Therefore, a potential explanation of the heightened query frequency observed in the continuous model could be that user specifically formulate new queries during rechecking. However, such queries would most likely be more specific than the ones issued earlier in the session. Since this information is not included in the modelling process in its current form, the findings need to be confirmed in a follow up study, for example, by adding query length as an additional feature to the model. However, such richer representations would need to utilise the extended expressiveness of multi-space HMMs or related techniques.

The continuous model was also used to analyse the search phase transition point based on the entirety of available search sessions. Overall, approximately 37% of the sessions are transitioning to the second search phase at some point during their persistence while phase transition, if at all, is occurring independently from session length. The large proportion of non-transitioning session is believed to be caused in the uncontrolled and heterogeneous nature of the underlying data including different task types and information needs.

Lastly, the continuous model is also used to exemplify how a phase-based hidden Markov model of users' search behaviour could be used to estimate some of the

crucial parameters of the interactive probability ranking principle. In this context, it could also be shown that estimates of the expected time until the next relevant document is found are best when the users' current search phase is modelled on a continuous scale.

Despite the fact that the models were built successfully, usage of unlabelled training data is posing a great challenge in practice since the effect of task type might superimpose the effect of search phase with respect to the parameters in question. Therefore, ideally models should be trained on a suitable controlled dataset before being tested on uncontrolled data. In fact, Kotzyba et al. (2017) have demonstrated that HMMs are in principle able to classify search session types when models are being training on data generated in a controlled lab environment. It is worth exploring in the future whether a similar process could also be employed for search phase modelling by using means already present in the framework.

Apart from better controlling confounding variables in further experiments, the presented models could be improved further by a few other factors. First, current models use implicit relevance signals as an approximation to model phase transition. In reality phase transition is likely to occur at various additional points in the search process and the models presented here are unable to capture this. Furthermore, phase transition is modelled to be one-directional only. However, in practice users may experience a topical drift getting sidetracked in their research by newly discovered knowledge (Sadikov et al., 2010). An ideal multi-stage search phase model would also be able to determine such shifts and consequently reassign users in Searching phase. However, even when retaining the general semi-Bakis structure, models could be extended to include additional search phases to match cognitive models more closely.

Overcoming the afore mentioned challenges and limitations will benefit the researcher in the field in at least two ways. On the one hand, as was shown earlier, hidden Markov user models can principally be used to finally determine the parameters of the interactive probability ranking principle, which could not be achieved by using cognitive models in the past. On the other hand, holistic modelling of user behaviour can be instrumental to the creation of adaptive search interfaces that provide situation specific assistance to further increase search effectiveness and efficiency in complex search tasks.

CHAPTER 5

Analysis of Potentially Rumourous Twitter Conversations

The second application example of the HMM framework focusses on rumours and how they are distributed through social media. By using Twitter data the framework it not only applied to different kinds of features originating from another source compared to the first application. More importantly, in this application the framework is utilised to build a classification system. Moving the focus from manual to automated model interpretation also allows exploitation of the sophisticated multi-space HMMs for the task, which can be used to combine continuous with discrete features to create a unified model.

Not least because of a massive trend of political instrumentalisation of social media channels in recent years nowadays *misinformation* and *fake news* are largely recognised as major problems. Closely related to these phenomena are *rumours* which are prevalent in social media as fast-spreading and unverified pieces of information (Zubiaga et al., 2018). While the term fake news has been used rather loosely as a catch-all phrase in the past, the main distinction of fake news and rumours is that former are purposefully created with malicious intent to deceive the public, as such fake news are by definition false information (Meinert et al., 2018). The term rumour refers to information that is carelessly distributed without being checked for its factuality but not necessarily motivated by the intention to deceive. Rumours differ from fake news especially since they can in the end turn out to be true, while they may also be false or remain unverified indefinitely.

Twitter as the primary platform for real-time news (Hu et al., 2012) and unmod-

erated in nature ideally facilitates the spreading of rumours (Qazvinian et al., 2011). This fact is for example further enhanced by the *retweet* feature, where a user can distribute information with a single click effectively broadening the rumour's reach. However, the distribution of rumours is also often caused by users' ineradicable personality traits, for example, the need for attention or heightened levels of anxiety (Jaeger et al., 1980). Furthermore, in many cases social media participants' laziness prevents them from adequately scrutinising pieces of information before sharing it with others.

In an attempt to curb rumour spread, especially methods for automated rumour detection and veracity classification have become popular research topics. In the survey paper by Zubiaga et al. (2018) the authors propose a general pipelined architecture of an automated rumour veracity system which includes four steps (Figure 5.1): (1) performing rumour detection on a stream of social media posts, (2) tracking responses to newly discovered rumours, (3) tweet level rumour stance classification for tracked responses and finally (4) determining the actual truth values (veracity) of the rumours.

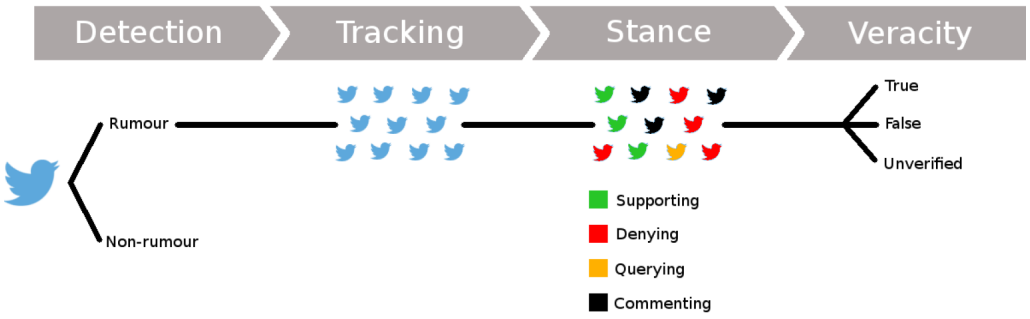


Figure 5.1: Rumour veracity classification pipeline by Zubiaga et al. (2018)

Naturally, the overall goal of the proposed system is the final veracity judgement and research so far has often solely focussed on the last of the four steps. In contrast, the tracking of responses has received very little attention in the literature so far, while one notable example of advancements in this direction is described by Hamidian and Diab (2015). Additionally, rumour stance classification has been subject of research in the past, for example, in the work by Procter et al. (2013). Following their methodology the stance of a tweet with respect to a rumour can be either *supporting*, *denying*, *questioning* or *commenting*.

Additionally, Zubiaga et al. (2016) showed in their analysis of rumourous conversations that stances could be indicative of rumours' veracity as they show distinct temporal patterns with respect to the veracity value. Nevertheless, stance is underused in related work as most of the approaches for veracity classification use feature heavy natural language processing methods (Castillo et al., 2011;

Kwon et al., 2013; Vosoughi, 2015; Yang et al., 2012). Ma et al. (2017), Wu et al. (2015) and Lukasik et al. (2016) extend on this by also regarding the progression of features over time. Some of the few exceptions where stance is used as an additional feature are the papers by Liu et al. (2015a) or Enayet and El-Beltagy (2017). However, none of the prior works focussed on the power of stance as the only feature for veracity classification.

Prior related work on the detection of rumours has often focussed on rumours known a priori (Hamidian and Diab, 2015, 2016; Qazvinian et al., 2011). This is contrasted by Zhao et al. (2015), who made use of manually curated keyword lists. While the first approach is unsuited for detection emerging rumours, the second approach suffers from insufficient recall in the detection task. Up to this point, there is no prior work that uses stance for the rumour detection task. The evident need for additional means for rumour detection and veracity classification is also emphasised in Zubiaga et al. (2018).

While response tracking and tweet stance classification are out of the scope of this thesis, the remaining two crucial steps in the rumour pipeline—rumour detection and veracity classification—are tackled from a novel direction in this second application of the HMM framework. By focussing on sequences of *tweets' stance* observations used alone and as a joint feature together with the tweets' posting times, crowd wisdom is exploited to complete both tasks. The main hypothesis is that as rumours evolve over their lifetime so do the stances expressed towards them in the responding users' tweets. Therefore, these temporal patterns in the tweet sequences act as the primary classification feature not only to distinguish rumourous conversation from non-rumourous but especially to predict rumour veracity.

Abstracting from the tweets' textual content to considerations on stance level is theoretically beneficial on at least two levels. First, it can be argued that tweet content is not always a useful feature towards the classification tasks as it simply could be misleading or wrong. However, more importantly, concentrating on one feature also makes an eventual automated veracity classification system easier to realise. In fact, stance has already been obtained automatically from tweets with reasonable accuracy in the literature (Aker et al., 2017a,b). In contrast, feature heavy natural language processing approaches also require provision of features that are more costly in acquisition, e.g. social features which require scanning of the message graph.

Rumour tracking and stance classification as the intermediate stages in the veracity detection pipeline are not part of this thesis. Instead an existing dataset extracted from Twitter is used which comprises rumours and also a large number of non-rumourous conversations. The rumours are annotated for their veracity and a portion of the data is also annotated for stance on tweet level which acts

as gold data in the following experiments. In further experiments gold annotations are replaced and supplemented by automatically generated labels to increase dataset size as well as to tackle the rumour detection task. Moving to automatically generated labels not only demonstrates the stability of the classifier in terms of F1-scores, but also approximates the application of the system on a large scale where manually generating labels becomes infeasible.

Furthermore, additional experiments are conducted, investigating the explanatory power of the stance feature in combination with the tweets' posting times. Timely detection and classification of emerging rumours will be one of the key success factors of any eventual productive system. Therefore, this is also explored in an experimental setting by deliberately limiting input sequences' length and comparing classification performance to a classifier using all available information.

The remainder of this section is structured as follows: First the dataset used is introduced in the next paragraphs, before describing the necessary preprocessing procedure and the hidden Markov modelling parameters used. Afterwards, the results of applying the framework to both tasks are presented, starting with rumour detection and followed by veracity classification. The section closes with a discussion of the results and an outlook on further research in the area.

5.1 Generating Datasets

All experiments regarding rumour detection and rumour veracity classification described in the following sections are based on data gathered from Twitter. Specifically, subsets of the dataset created by Zubiaga et al. (2016) are used as they are appropriate for the respective application. The source dataset was released under public domain and is the only dataset containing rumour veracity annotations as well as stance information on tweet level which is publicly available. In total, it consists of 7507 conversation threads concerning nine different events that took place between August 2014 and March 2015.

Based on the method how the data was acquired, the nine events can be divided into two principal categories. On the one hand, five events are *breaking news* which is in this context defined as events that are likely to spark the distribution of multiple rumours across social media channels. However, which rumours actually were propagated in the context of these breaking news events was unknown a priori. Instead, conversations concerning these breaking news events had be to tracked manually, isolating those conversations that attracted the most attention in terms of retweets and replies and adding them to the dataset. The remaining four events in the dataset are specific rumours which were known a priori. Conversations regarding these events could be tracked more easily by using specific

keyword searches using Twitter's streaming API.

As motivated above, the eventual goal is to create an automated rumour veracity classification framework for emerging and fast spreading rumours. Given the definition of breaking news events as created by Zubiaga et al. this event type exactly matches the situation where automated veracity classification will be most beneficial. Therefore, the following considerations mainly focus on the breaking news events, which are introduced briefly below:

- *Charlie Hebdo attacks*: The Paris main office of the French satirical newspaper Charlie Hebdo was attacked by terrorists killing and wounding numerous people.
- *Ferguson riots*: After an incident of police violence in Ferguson, Michigan, USA social unrest broke out in the region.
- *Germanwings plane crash*: A commercial plane was deliberately crashed into the Alps by the co-pilot killing all people on board.
- *Ottawa shooting*: A Canadian soldier was shot in Ottawa, Canada.
- *Sydney siege*: A hostage situation in a café located in central Sydney, Australia.

Figure 5.2 details an example of a rumourous Twitter conversation regarding the Ferguson riots breaking news event, which sparked especially many false rumours.

All 7507 conversation threads were classified and annotated as either being rumourous or non-rumourous by journalists who all were familiar with the respective events. During the annotation process a total of 2695 conversations was deemed as rumourous and the remaining 4812 as non-rumourous. Furthermore, rumours were also annotated for their veracity which can be either *true*, *false* or still *unknown* by the time the annotations were generated. However, due to budget constraints the set of rumourous conversations was sampled randomly, finally resulting in only 330 rumourous conversations being annotated for veracity. It is worth noting that 302 of the 330 randomly selected rumourous conversations were held with respect to one of the breaking news events since these events overall caused a far larger response across the Twitter community.

The sampled 330 conversations additionally were annotated on tweet level for their stance regarding the respective rumour via a crowdsourcing procedure. Following the definition of stance categories as introduced by Procter et al. (2013), the possible annotations for each tweet are *supporting* for all tweets where the tweet's author was agreeing with the rumour source tweet, *denying* for tweets expressing disagreement with the source, *questioning* for tweets that convey an

Figure 5.2: Excerpt of a Twitter conversation about the Ferguson riots event

inquiry about additional information concerning the rumourous statement and finally *commenting* for all remaining tweets that do not express clear agreement, disagreement nor inquiry.

The dataset was processed as part of a structured and guided annotation procedure which also included additional information about the annotation process itself, like the annotators' confidence in their own judgements. Furthermore, detailed information on rumour as well as on tweet level is included in the dataset, e.g. the type of rumour (misinformation vs. disinformation) or information about the presence of any evidence supporting the claim.

Apart from the annotations, several meta data pieces are given in the dataset which were directly extracted from Twitter during the crawling process. These include tweet level meta data, e.g. time of tweets' creation, retweet count or indication of the presence of hastags or URL as well as user level information, e.g. account creation date or follower count amongst many others. Tweet meta data and the set of annotations are stored together with the tweet itself in JSON format in a single file for every tweet. All JSON files are arranged in a directory structure indicating the tweets' dependencies.

This dataset's usefulness as a basis for the second application of the HMM framework is justified at least twofold. On the one hand, because of its rich annotations, it allows the direct exploitation of user behaviour captured in form of the stance

feature for the rumour detection and veracity classification tasks without the need for further data acquisition steps. On the other hand, the dataset has also been used in a number of related work, including e.g. the 2017 rumour veracity classification challenge SemEval[1]. Therefore, usage of this dataset allows a direct comparison of the methods developed in the scope of this thesis with state-of-the-art work from the literature proving valuable insight about the HMM-based classifier's performance.

In the following sections, three variants of the dataset are introduced and labelled which are tailored to the two classification tasks as well as to the specific research question investigated in the respective experiments.

5.1.1 Dataset $detection_{auto}$ for Rumour Detection

Dataset $detection_{auto}$ is the most comprehensive of the three datasets since it is the only set that contains rumours as well as non-rumours. Therefore, this dataset is applicable to the rumour detection task. Since the initial dataset does not contain tweet level stance annotations for the non-rumourous conversations, these labels were generated using the stance classifier by Aker et al. (2017a). The stance classifier performs standard feature engineering directly on the tweets' content—which is available for all conversations—and can be executed without setting any parameters. After generating all missing stance labels, very short conversations were excluded from the dataset. Since in the rumour detection task a timely classification of emerging rumours is a central aspect, the minimal sequence length is set to five tweets. Table 5.1 details the distribution of rumours (R) and non-rumours (NR) across the five breaking news events.

Overall, a class imbalance towards non-rumourous conversations can be observed while rumours constitute only close to one third of the data. On event level it is apparent that two of the five events sparked more rumourous than non-rumourous conversations opposing the overall trend. However, these two events were also inducing the fewest conversations. Furthermore, it is worth noting that this dataset contains conversations that are marked as rumours while their veracity value itself is unknown. This is the case for all conversations that were originally not included in the set of conversations sampled for manual annotation and does not imply that their veracity is actually unrecognisable.

[1] `http://alt.qcri.org/semeval2017/task8/`—last accessed 24.10.2018

Table 5.1: Dataset $detection_{auto}$ including rumourous (R) and non-rumourous (NR) conversations

Event	Conversations	R / NR
Charlie Hebdo Attacks	1735	368 / 1367
Ferguson Riots	915	240 / 675
Germanwings Plane Crash	238	122 / 116
Ottawa Shooting	671	361 / 310
Sydney Siege	1045	438 / 607
Total	4604	1529 / 3075

5.1.2 Dataset $veracity_{gold}$ for Rumour Veracity Classification

Additionally, two datasets for the veracity classification task were build. The first dataset variant $veracity_{gold}$ comprises all rumourous sequences from the breaking news events where tweet level stance annotations generated by human annotators are available. As motivated earlier, restricting this dataset to breaking news events best approximates the general purpose of the classifier—being applied to unknown and emerging rumours which are unknown a priori. The reduced set of 302 conversations was then further filtered to include only sequences with a length of ten or more tweets to ensure that the classification tasks can be performed on the basis of a sufficient observation count. Furthermore, all rumourous sequences were excluded from this dataset whose veracity was still unknown at annotation time, since this dataset was intended to be applied to a binary veracity classifier only.

Combining these filtering steps reduces the size of the dataset to 173 sequences with a length of ten or more tweets from the five breaking news events (Table 5.2). Naturally, since the data was gathered based on real events, some of the events spawned more conversations meeting the criteria described above than others, ranging from twelve conversations regarding the Germanwings plane crash to fifty conversations with respect to the Sydney hostage situation. It is also worth noting that the dataset is imbalanced on event level regarding rumour veracity. Especially, the Ferguson riots and the Germanwings plane crash sparked strikingly more false than true rumours. However, overall the dataset's balance is satisfactory with a ration of approximately 47% to 53% for true and false rumours. This dataset is fundamental to the main experiments investigating stance-based veracity classification.

Table 5.2: Overview of dataset $veracity_{gold}$ including rumours with at least 10 tweets

Event	Rumours	True / False
Charlie Hebdo Attacks	46	24 / 22
Ferguson Riots	34	2 / 32
Germanwings Plane Crash	12	2 / 10
Ottawa Shooting	31	20 / 11
Sydney Siege	50	33 / 17
Total	173	81 / 92

5.1.3 Dataset $veracity_{auto}$ Utilising Automatically Generated Stance Labels

Dataset $veracity_{auto}$ is an adaptation of $veracity_{gold}$ where all manually generated stance labels are replaced by automatically generated labels. For this purpose again the stance classifier by Aker et al. is used. In this context it worth noting, that the stance classifier has also been tested on rumourous conversations and achieved the best results in the *RumourEval 2017 shared task A* challenge with an accuracy score of 0.790. In the RumourEval challenge the organisers also used the rumour dataset created by Zubiaga et al.—although a different sample of it. It is therefore reasonable to assume a stance classifier performance in the range of 80% accuracy on dataset $veracity_{gold}$ as well. Otherwise dataset $veracity_{auto}$ is identical to $veracity_{gold}$, i.e. it comprises the same events, rumours and tweets.

5.1.4 Preprocessing

Before conduction the classification experiments, all datasets had to be processed for compatibility with the modelling framework. Therefore, the data was converted from its original JSON format to a set of observation sequences suitable for HMM-based modelling. To this end, the order of operations was as follows:

Foremost, since the goal of this application is to investigate stance as the main feature for rumour detection and veracity classification, all tweets' textual information as well as most of the meta data is discarded from further analysis. Therefore, in the following a tweet T is defined as a tuple $T = (t_{stance}, t_{time})$, where t_{stance} is the tweet's stance annotation and t_{time} the tweet's time of creation. Furthermore, the conversations' hierarchical thread structure (compare Figure 5.2) is collapsed to a one-dimensional sequence of tweets in chronological order. Figure 5.3 visualises the representation of a conversation as a sequence of

stances. In the next section the method for creating HMM-based classification systems is introduced.

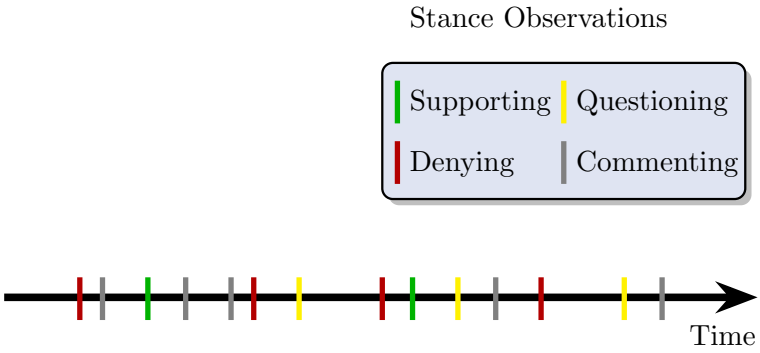


Figure 5.3: A Twitter conversation represented as a sequence of stances

5.2 Generating the Classifiers

Both classification tasks were performed using two principal modelling setups which are introduced in this section. Thereby the main difference is that the first approach used stance as the only feature to model conversation properties whereas in the second approach the time of posting was also included as an additional feature. In both variants unconstrained hidden Markov models were trained to best fit the respective classes' characteristics and subsequently used to build the actual classifier.

5.2.1 Using Stance as the Only Feature

The first setup uses discrete ergodic HMMs to model stance as the only feature which is in the following referred to as hidden Markov model θ. Especially, this model makes no use of the tweets' posting times apart from determining the order of the sequence of stances. Therefore, rumourous and non-rumourous conversations are converted to a succession of stance values as depicted in Figure 5.4. Model θ also acts as an additional baseline to the posting times-aware model introduced below, by isolating the effect of including the tweets' times as a feature on the performance of the models.

Since the observation alphabet E of model θ consists only of the four possible stance values, this setting can easily be described with discrete HMMs. An example of this configuration using three hidden states can be seen in Figure 5.5.

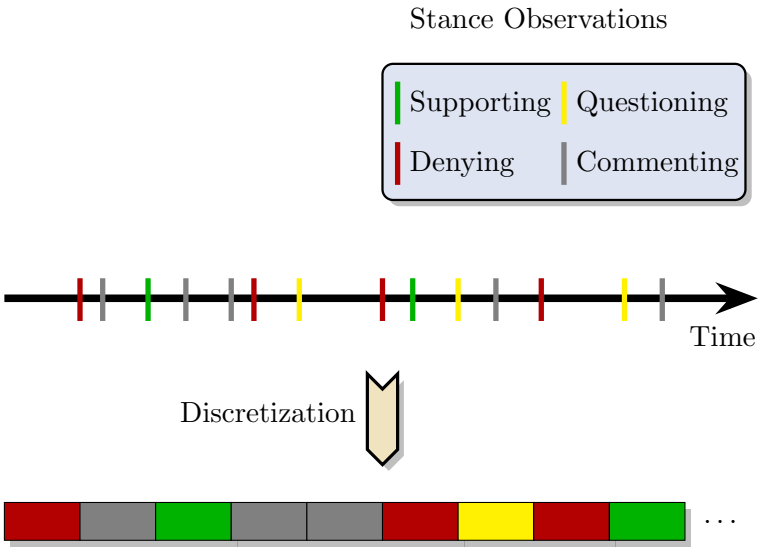


Figure 5.4: Visualisation of an input sequence used by the discrete models

Unlike in the search phase modelling application, however, there exists no prior knowledge about the hidden process described in this model. Especially, no possibility is known to a priori determine any of the hidden state transition or emission probabilities. Therefore, the only reasonable initialisation is to assign a random stochastic matrix for both the transition probability matrix A and the emission probability matrix B.

Furthermore, the set of hidden states most appropriate for the tasks has to be determined, which is one of the HMM-related problems for which no general solution can be specified (Rabiner, 1989). Since there also exists no related work on building a Twitter conversation classifiers using HMMs, the hidden state count N has to be determined empirically. Successively calculating the models with increasingly larger N and evaluating results using appropriate metrics, the best performing model sizes are found to be in the range $N = [3, 15]$ hidden states. In the following experiments, the final hidden state count is determined individually depending on the task, dataset and the feature variant used.

Additionally, the start vector π is assigned at random before using ten iterations of the Baum-Welch algorithms to tune all model parameters to the training data. To overcome the problem of suboptimal start value configurations causing the EM-algorithm to get stuck at poor local optima, one thousand unique random start value configurations are optimised for every hidden state count N. Finally, only the best performing models are kept as part of the respective classification systems.

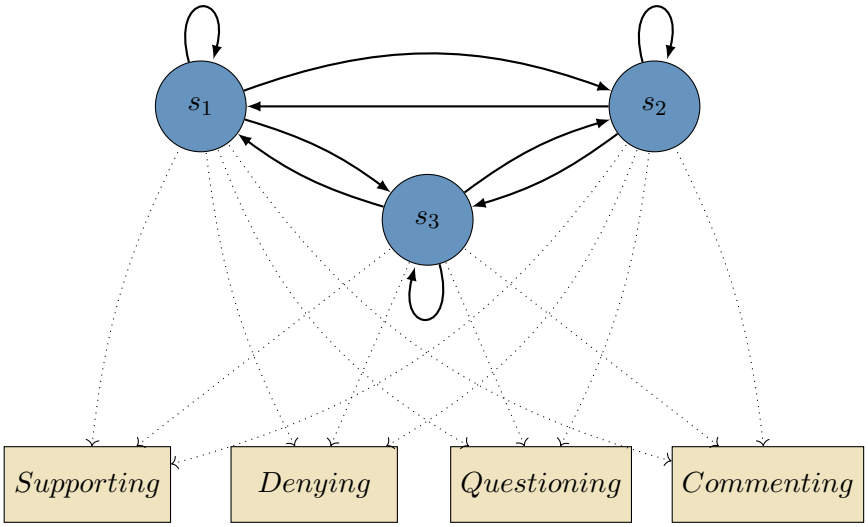


Figure 5.5: Example of a discrete HMM using three hidden states

5.2.2 Using Stance and Tweet's Posting Time as Joint Feature

The second modelling variant denoted as θ' utilises the full potential of the information contained in the tweets' stance and the tweets' posting times for the classification tasks by using multi-space hidden Markov models. These models are the most versatile variant of the HMM family, effectively allowing to unify discrete (the tweets' stances) and continuous (the tweets' posting times) observations as a joint feature (Chapter 3). This effectively resolves the need to introduce any constraints with respect to the underlying hidden process. The basic configuration of HMMs θ' regarding the transition probability matrix A the start vector π and the hidden state count N remains unchanged compared to model θ. Furthermore, Baum-Welch parameter optimisation is performed using ten iterations based on one thousand random start value configurations for every $n \in N$ while finally only retaining the best performing models.

The major distinction of models θ' is its richer definition of the observation alphabet $E = \{(t_{stance}, t_{time})\}$ and the usage of multiple spaces to be able to model the tweets' times jointly with the tweets' stances. However, following the formal definition of multi-space HMMs, the posting times need to be specifiable by a function satisfying the constraints given in Equation 2.12. Therefore, it is necessary for the absolute timestamps, which are specifying date and time, to be converted to a continuous range of values. This can be accomplished by introducing a conversation specific time line defined for each thread, which is initialised with the timestamp $t_{time} = 0$ for each conversation's starting tweet.

All responding tweets' timestamps are converted to numerical values using two principal methods: First, t_{time} of each response is converted to represent the seconds elapsed since the respective conversation was started by the first tweet. Following this conversion method the beginning of a sequence s could for example be represented as

$$s = (Supporting{:}0, Commenting{:}180, Denying{:}420, Commenting{:}540, \ldots).$$

Alternatively, as a second method timestamps of the responding tweets are transformed to indicate the seconds elapsed since the *directly preceding* tweet was posted in the same conversation. Using this conversion the same sequence is represented as

$$s' = (Supporting{:}0, Commenting{:}180, Denying{:}240, Commenting{:}120, \ldots).$$

Conversations' length shows large variance, both when comparing conversations regarding the same event as well as between events' average conversation length. For example, the average conversation length in dataset $veracity_{gold}$ is approximately 16 hours. However, most of the 180 conversations have below average duration while ten conversations continue for more than two days. This observation suggests that it could be beneficial to normalise tweet times. In the example, factoring out the effect of sequence length for sequence s while assuming a total length of one thousand seconds would—depending on point of reference used—respectively transform the sequences as follows:

$$s = (Supporting{:}0, Commenting{:}0.18, Denying{:}0.42, Commenting{:}0.540, \ldots)$$
$$s' = (Supporting{:}0, Commenting{:}0.18, Denying{:}0.24, Commenting{:}0.12, \ldots).$$

Given both points of reference for the tweets' times and the possibility of employing normalisation, in total four potential mappings are considered for model creation. In principal, all mappings comply with the general theory underlying this modelling approach by utilising the information contained in the tweets' posting times as a joint feature for the classification tasks. Therefore, performance scores of θ' models are expected to exceed those of θ configuration models, regardless of how the times are converted. However, conducting full size experiments employing all different transformations considerably increases calculation complexity. Therefore, in a pilot study limited to models with ten hidden states and using only one hundred random starting parameter configurations all conversion methods were compared with respect to their effect on the final classifier's performance. Hereby, the conversion of tweets' times with respect to the conversations' starting tweets shows better results. Normalisation of times does not increase the performance. Therefore, in all following experiments classifications are based solely on the non-normalised representation of tweets' posting times.

Figure 5.6 visualises the conversion of a Twitter conversation to a succession of joint stance and time observations for multi-space HMM-based modelling.

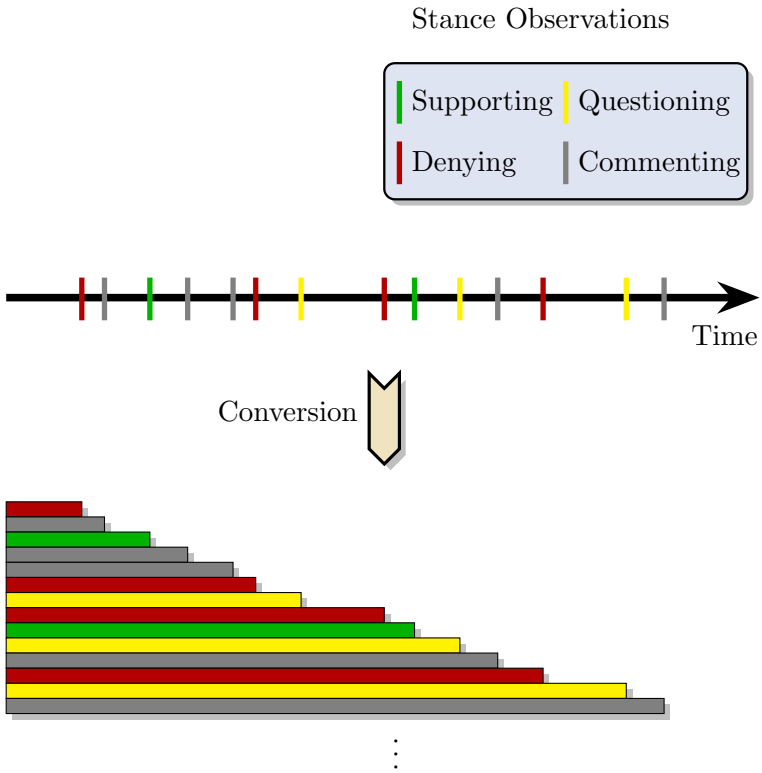


Figure 5.6: Visualisation of an input sequence used by the multi-space models

The last component of model θ' is the emission probability matrix B, which is set up for the multi-space setting. Given the four stances each is assigned its own one-dimensional real space $\Omega_\sigma = R^{1\sigma}$, where space weights w_σ are determined by summing stance σ's occurrence count in the training data and dividing it by the total number of observations. The spaces' probability density functions $N_g(x)$ are initialised as exponential decay functions. Following the assumption that conversation activity subsides with time, the decay rate is initially set to 1 yielding a final function of the form $N_g(x) = e^{-x}$. Space weights and probability density functions are initially assigned uniformly across all hidden states before optimising the functions' rate as well as all space weights using the Baum-Welch algorithm adapted for multi-space HMMs. An example of the multi-space model using three hidden states is given in Figure 5.7. It is worth noting at this point that although alternative probability density function configurations are possible, changing the functions' properties has only minimal effect on the final classifier's performance.

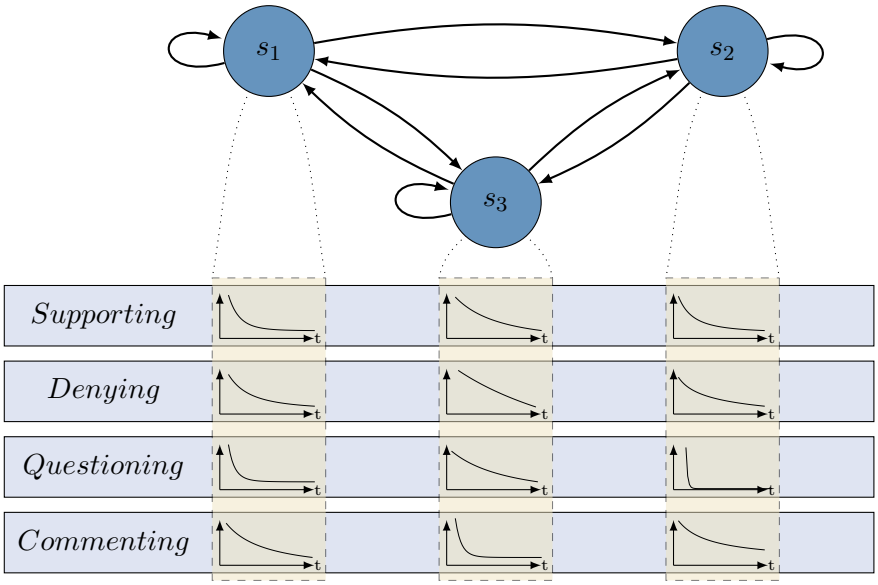


Figure 5.7: Example of a multi-space HMM using three hidden states

5.2.3 Class Assignment Formula

Regardless of task, both HMM variants introduced above can be used as a binary classifier. To this end, two identical copies are initialised at first. Afterwards, training data is split into two subsets given the two possible class values (true and false condition) and each subset is used to tune the parameter of one of the model. The outcome is two HMMs θ_{true} fitted to true condition and θ_{false} fitted to false condition, which are utilised to determine the class label of any conversation ε_i in the test set following Equation 5.1.

$$C(\varepsilon_i) = \underset{c \in \{false, true\}}{\operatorname{argmax}} \; \mathbb{P}(\varepsilon_i | \theta_c) \tag{5.1}$$

In this expression, sequence probability $\mathbb{P}(\varepsilon_i|\theta_c)$ can be calculated by using the forward-part of the Forward-Backward-Algorithm for HMMs (Rabiner, 1989). Because of small sequence probability values associated with long observation sequences, all calculations are performed in logarithmic space to avoid underflows.

Considering Equation 5.1 it is worth pointing out that the conditions' prior probabilities (for example proportion of rumours vs. non-rumours) are not taken into account when making the class predictions. In principle, the classifier could be made aware of the prior probabilities by using Bayes' rule and determining class label following Equation 5.2.

$$\mathbb{P}(true|\varepsilon_i) = \frac{\mathbb{P}(\varepsilon_i|true)\mathbb{P}(true)}{\mathbb{P}(\varepsilon_i|true)\mathbb{P}(true) + \mathbb{P}(\varepsilon_i|false)\mathbb{P}(false)} \tag{5.2a}$$

$$\mathbb{P}(false|\varepsilon_i) = \frac{\mathbb{P}(\varepsilon_i|false)\mathbb{P}(false)}{\mathbb{P}(\varepsilon_i|true)\mathbb{P}(true) + \mathbb{P}(\varepsilon_i|false)\mathbb{P}(false)} \tag{5.2b}$$

However, when applying the classifier in practice prior probabilities are unknown. Given that already the five breaking news events considered in the experiments show clearly different prior probability ratios regarding both target classes, tuning the classifier to these prior probabilities would diminish its generalisability and is therefore omitted.

In the next section, the results of applying the HMM classifiers to the rumour detection task are detailed, investigating one of the central prerequisites for rumour veracity prediction.

5.3 Subtask I: Rumour Detection

As motivated earlier, the prerequisites for realising an automated rumour veracity classification system are that rumours are identified, tracked and extracted from social media channels first. Only in combination with these steps the actual rumour veracity classification can be performed in an eventual application. While rumour tracking and feature extraction are out of the scope of this thesis, the rumour detection task has already proven to be a substantial challenge in related work and is further investigated in the following.

In principle there exist two approaches to detect rumours on Twitter. The first approach makes use of manually curated lists of keywords which are believed to be indicative of a rumourous conversation. An example of this approach is the work by Zhao et al. (2015) where the authors use regular expressions on Twitter's trending topics for this task. However, performance of their method varies substantially in terms of precision and remains untested for recall. Additionally, automated methods can be used for the task as it is, for example, demonstrated by Zubiaga et al. (2017), who use conditional random fields. In their work the authors exploited tweet content-based features alongside social features obtained from tweets' meta data and achieved an F1-score of 0.607 on the breaking news dataset, efficaciously creating the state-of-the-art for rumour detection on Twitter.

Although the CRF-based rumour detector is state-of-the-art in terms of F1-score, it still shows potential for improvement regarding its recall with a score of 0.556.

This also becomes apparent when considering the performance scores of the baselines referenced by Zubiaga et al. (2017), where already a Naïve Bayes solution achieved a recall of 0.723.

When evaluating a classifier, the cost of different types of misclassification also can be taken into account, for example, if the data has a strong class unbalance or if the application imposes a particular interest in one of the classes (Witten et al., 2016, Chapter 5.8). Rumour detection can be viewed as an instance of these problems since missing a particular rumour due to insufficient recall of a model is more costly than erroneously flagging a conversation as rumourous. In the former case, a rumour will continue to develop and spread through the social media channels potentially causing all unwanted negative effects. Unless the classifier is re-run on the same conversation at a later point in time, there remains no additional mean of detect that it is a rumour. The opposite case of falsely flagging a conversation as rumourous could also cause irritation—especially when making the false judgement transparent to the conversation participants. However, detected presumable rumours will also traverse farther through the veracity classification pipeline making it more likely that the erroneous classification will be detected and corrected at a later stage.

Since current automated veracity classification methods are still not operating flawlessly, in a critical situation rumour veracity may remain to be checked manually by experts. However, manual assessment is costly and can not be performed on the entirety of social media conversation, even when being restricted to a specific topic or event. Both these considerations motivate the creation of a high recall rumour detector that acts as a *filter* for the later stages of the pipeline—regardless of whether manual or automatic means are used to finally determine the veracity. While achieving a very high recall inevitable will lead to a decrease in precision, the primary goal of such a system is to not miss rumourous conversations.

Hidden Markov models have shown in the past to be well suited for high recall tasks—especially when compared to CRF approaches on the same task. In the work by Ponomareva et al. (2007), HMMs and CRFs are compared in named entity recognition for medical texts on more than 100,000 named entities. The authors found that HMMs have higher recall of $+4\ldots7\%$ depending on the type of entity, while overall CRFs show better performance in F1-score (0.687 vs 0.657 for HMMs).

Based on these considerations, the HMM framework is tuned for the rumour detection task. Overall two HMM variants for the features stance (model θ) as well as stance + time (model θ') are trained using the parameter setup and tuning procedure described in Section 5.2. Afterwards, rumour detection is performed using the class assignment formula detailed in Equation 5.1. Results are discussed

using dataset $detection_{auto}$ following the *leave-one-event-out* cross-validation procedure.

The leave-one-event-out cross-validation setup used for the following evaluations is adopted from the work of Lukasik et al. (2015) where the authors perform stance classification on tweet level given a set of rumours. In this setup, the models are trained on all rumours originating from four of the five events while the rumours of the remaining event are used for testing. The procedure is repeated five times resulting in individual test scores for all events. These scores are subsequently used to create the single overall evaluation score as a weighted average across all runs.

The main strength of this approach is that it constitutes a realistic evaluation scenario in the context of social media rumours, by setting aside the conversations from an entire event during training to be later used for evaluation purposes. Therefore, the resulting evaluation scores simulate a practical application of the tested method in the real world, where a current event is resulting in emerging rumours that were not available during model training. Furthermore, taking into account that all events have individual properties that are inevitable learned by the classifier, the leave-one-event-out cross-validation is also the harder task to achieve acceptable scores in since no knowledge about the unseen events' properties is obtained during training. Although the alternative approach of tuning the classifier while considering conversations from all available events would presumable lead to better scores, it would impair the generalisability of results by making information available to the classifier that is not present when actually applying the method to new data.

In the following, F1-scores are used as the primary measure for all leave-one-event-out evaluations. Furthermore, they are used as the main criterion for selecting the best performing models across the candidate set created during the model generation phase. Choosing F1-scores as classification quality measure over other possible scores (for example accuracy) is appropriate because of the class imbalance in the data.

Both feature selections are evaluated under varying conditions, each focussing on different aspects of the task. First, the overall quality of the classifiers is investigated using all available conversation contained in dataset $detection_{auto}$, in which only those conversation were excluded that attracted less than four replies. Additionally, rumour detection is evaluated under the aspect of timeliness by varying sequences' minimal length. Finally, the predictive value of the individual stance labels is examined, exploiting the fact that the rumour detection dataset is larger than the other datasets due to the usage of automated stance classification. In the following section, results of these experiments are detailed beginning with the overall classification scores.

5.3.1 Overall Classification Results

When performing leave-one-event-out cross-validation for the five breaking news events using all 4605 available conversation in dataset $detection_{auto}$, system θ achieves an F1-score of 0.516 in the rumour detection task, with a precision of 0.355 and a recall of 0.948 for the target class *rumour*. In case the rumourous property is modelled using the joint feature of stance and time (model θ'), the performance increases to an F1-score of 0.568 with a precision of 0.416 and a recall of 0.893 (Table 5.3a).

Despite the high recall, both models fall short of the current state-of-the-art conditional random field introduced by Zubiaga et al. (2017), which achieved an F1-score of 0.607. However, when directly comparing the models, it has to be noted that the authors are using in total 5802 conversations originating from the five breaking news events, since they are introducing looser sequence inclusion requirements. As it will become evident in the following sections, test and training dataset size as well as individual properties of included conversations affect overall classification performance. Therefore, the CRF has additionally been retrained and tested on dataset $detection_{auto}$. Interestingly, while the model is experiencing a two percentage point decrease in precision it simultaneously shows a two percentage point increase in recall. As a result, its performance in terms of F1-score remains unchanged (0.607) on the new dataset.

Given the evaluation procedure, classification results of HMM-based classification can also be split into the five individual runs (Table 5.3b–f). Looking at the detailed results, it becomes obvious that both systems have a bias towards classifying conversation as rumourous. Consequently, the recall of model θ ranges from 0.832 to 1.0 while system θ' shows a slightly worse recall ranging from 0.671 for event Charlie Hebdo attacks (5.3b) to a recall of 1.0 for the Germanwings plane crash (5.3d). The latter event is also the only event where the multi-space HMM has higher recall than its discrete counterpart.

Considering the precision of both models their performance is far less impressive. Here, both models show a similar score distribution, clearly performing worse in conversations related to the Charlie Hebdo attacks (θ: 0.233 vs. θ': 0.333) and the Ferguson riots (θ: 0.270 vs. θ': 0.311) compared to the other three events. Looking at the distribution of rumours and non-rumours (Table 5.1), it is striking that the models can achieve better performance when events show a more balanced distribution of class labels, which is the case for the remaining three events (Table 5.3d–f). In this case, models' precision ranges from 0.426 to 0.539 for system θ and 0.443 and 0.544 for system θ' respectively. Besides, it can be seen that θ' gives more precise results for all events and hence has the overall superior F1-scores. When considering these detailed figures, it becomes apparent that overall θ and θ' fall short of the CRF because of their insufficient precision.

Table 5.3: Rumour detection performance across five breaking news events

(a) Overall classification performance

System	Precision	Recall	F_1
θ	0.355	**0.948**	0.516
θ'	**0.416**	0.893	**0.568**

(b) Charlie Hebdo Attacks

System	Precision	Recall	F_1
θ	0.233	**0.832**	0.364
θ'	**0.333**	0.671	**0.445**

(c) Ferguson Riots

System	Precision	Recall	F_1
θ	0.270	**0.971**	0.423
θ'	**0.311**	0.925	**0.466**

(d) Germanwings Plane Crash

System	Precision	Recall	F_1
θ	0.517	0.992	0.680
θ'	**0.530**	**1**	**0.693**

(e) Ottawa Shootings

System	Precision	Recall	F_1
θ	0.539	**1**	0.700
θ'	**0.544**	0.994	**0.703**

(f) Sydney Siegel

System	Precision	Recall	F_1
θ	0.426	**0.979**	0.594
θ'	**0.443**	0.947	**0.604**

5.3.2 Early Prediction of the Rumourous Property

Timeliness is one of the major requirements of a rumour detection system. In the previous experiment, this was accounted for by considering all sequences containing as few as five or more tweets for classification, since utilising the HMM to classify very short sequences is approximating the detection of emerging rumours. However, given the operating principle of the classifiers, they could be susceptible to performance deterioration when being constricted to short sequences. To gauge the extent of possible deterioration when performing the rumour detection task, the models are additionally evaluated on a subset of $detection_{auto}$. In this case, the 3513 sequences comprising at least ten or more tweets are selected which on average contain just more than 25 tweets. Using the modified dataset, models are retrained before performing three classification runs. First, conversations are classified based on all available tweets. Additionally, only the first ten tweets and only the first five tweets respectively are used to isolate the effect of early rumour detection. Since all three runs are based on the same conversations and model configurations the sequences' length at prediction time is the only variable.

Table 5.4 details the effect of conducting classification at specific conversation stages on both models' performance. A number of observations can be made based on these figures. At first glance, θ' appears to show a higher sensitivity to shorter sequences than model θ with F1-scores of 0.556 when using full sequences, 0.516 when using the first ten tweets and 0.503 when using only five tweets. θ on the other hand shows almost stable scores. However, θ generally has a lower performance level, even when using all available information with an F1-score of 0.508. Thus, in this experiment, the stance feature at full sequence length just barely outperforms five observations of the joint feature. Furthermore, taking a closer look at the individual predictions made by model θ, it can be established that this system fully deteriorates when input length is restricted, predicting $class = rumour$ for almost all test instances.

Table 5.4: F_1 scores when performing early rumour detection

System	All tweets	First 10 tweets	First 5 tweets
θ	0.508	0.492	0.492
θ'	**0.556**	**0.516**	**0.503**

This experiment also reveals another interesting observation when comparing the results to the prior experiment. Even when using all available information, both models' performance in this run is approximately one percentage point worse than when being trained on the entire dataset $detection_{auto}$. Considering that all other parameters remained unchanged, this observation suggests an effect of training data sampling on the overall results.

5.3.3 Predictive Value of Stance Labels

One of the particularities of hidden Markov models compared to most other machine learning approaches is that there exist no automatic means for feature selection. Instead, this decision remains to be made manually by the practitioner based on the specific application. In all previous experiments the stance feature has been compared to the joint feature of stance and tweets' times. However, stance itself has four discrete categorical manifestations, each of which with supposedly varying levels of relatedness to the target classification problems. For instance, it is reasonable to assume that observing supporting or denying stances is more meaningful for determining the veracity of a rumour than observing commenting tweets.

Following this intuition, an experiment was conducted where individual stance labels are purposely discarded from the sequences during model training as well as sequence classification. Since especially exclusion of commenting stances has a significant impact on sequence length—approximately 80% of the tweets are commenting—again a subset of sequences has been selected from $detection_{auto}$ where all remaining 1304 sequences meet minimum sequence length of five tweets under all conditions introduced below. This procedure allows effective isolation of the effect of stance label exclusion in the following experiment.

Commenting stance acts as the *null-label* during tweet stance assignment and thus is presumably conveying the smallest predictive value. Therefore, one experimental condition is created where all commenting tweets are removed for the data. In the following, this is denoted as SDQ, indicating the included stance labels. Furthermore, a second condition is created by additionally removing questioning stance leaving only the labels supporting and denying as features. This condition focusses on the supposedly most powerful labels and is denoted as SD. Since questioning stance is by far the least common label (only slightly more than 1% of all tweets) third condition SDC is defined, excluding this label from all sequences. The reasoning is that since only relatively few observations are available, the parameters associated with this stance label might not be optimised in the final models which could be impairing the performance of the classifiers. If this was to be the case, excluding this label altogether should increase performance. In Table 5.5 experimental results are given comparing SD, SDQ, SDC and $SDQC$ conditions in terms of F1 classification performance scores.

Generally, it can be seen that the models perform better when predictions are made based on all four stance labels. For model θ scores drop from 0.584 in condition $SDQC$ to 0.551 in SD. For model θ' a similar although less pronounced drop can be noted with scores ranging from 0.617 in $SDQC$ to 0.590 in SD. Noteworthy, also when individual stance labels are removed from the data, the joint modelling of stance and time usually gives better results, except for the

Table 5.5: F_1 scores for using different stance label selections

System	SD	SDQ	SDC	SDQC
θ	0.551	0.551	**0.599**	0.584
θ'	**0.590**	**0.603**	0.596	**0.617**

SDC condition where θ achieves a marginally better score of 0.599 compared to 0.596 for model θ'. SDC condition using model θ is also the only setting where discarding a stance label results in a performance improvement of 1.5 percentage points compared to using the full stance set. Comparing conditions SD and SDQ with $SDQC$, it can be seen that both models profit from modelling all stances. The joint model θ' also is impaired by removing questioning stance (SDC). The different rate of performance deterioration for both models again shows that feature selection has to be performed carefully and holistically including all other modelling parameters.

Given the results in Table 5.5, it is again interesting to observe that the selection of sequences also has a strong effect on the models performance. The $SDQC$ condition of this experiment is identical to the experiments above. However, solely due to sequence selection bias model θ shows an increase in F1-score of almost seven percentage points. The same is true for model θ' with an increase of almost five percentage points in F1-score. Since the remaining 1304 sequences used here due to the selection procedure are biased to feature an above average amount of supporting and denying stances, increased model performance makes it apparent that these labels are particularly helpful in determining the rumourous status. On the other hand, restricting models' features to those labels only (SD) does decrease prediction quality compared to condition $SDQC$.

5.3.4 Discussion and Future Work

Summing up the results of all rumour detection experiments, it is established that the HMM-based classification is best performed using the multi-space models making use of the joint modelling of stance and time. In this case, the hidden Markov model achieves an F1-score of 0.568 when using all 4604 conversations with five or more tweets as a data basis. The discrete variant of the HMM classifier achieves a score of 0.516 on the same task. However, both variants fail to beat the 0.607 F1-score of the state-of-the-art conditional random field classifier. When performance is considered in terms of precision and recall, however, it can be seen that the HMM-based classifiers also have particular strengths. Their recall of 0.948 for model configuration θ and 0.893 for model θ' surpass the recall of 0.574 achieved by the CRF by a large margin. In contrast, it is the hidden Markov

models' relatively poor precision that puts them behind the CRF in terms of F1-score. Especially, model θ produces imprecise results only performing slightly better than simply predicting $class = rumour$ for all conversations. Therefore, the following detailed discussion of results focuses on model θ', as it is clearly more suitable to determine conversations' rumourous property.

When striving for automatic rumour detection, timeliness of making class predictions is one of the key factors. It is also still a challenge for current models as can be seen by the score distribution when predictions are made based on successively shorter input sequences. Model θ' shows a clear drop in F1 performance from 0.556 to 0.516 when classification is performed after observing ten tweets. When input length is shortened further to five tweets performance decreases to 0.503. Therefore, it has to be noted that the HMM-based classifier is unable to determine the rumourous status of a conversation right from the beginning. This result does not come as a surprise given the fact that HMMs by nature best make use of sequential information. However, at the current time it remains unknown how the state-of-the-art CRF does perform when being restricted to very short sequences only and it would also be interesting to investigate this in the future.

In an attempt to boost overall classification scores, experiments were conducted to investigate on the predictive strength on the individual stance labels. In context of HMMs there exists no automatic solution to determine the usefulness of features. Instead, specific feature selections have to be tested manually for their influence on the quality of the resulting models. In the case of modelling conversations' rumourous properties only two facets of the tweets are used as features: stance and time. It has been established that inclusion of time as joint feature benefits the modelling process.

However, the inclusion of all individual stance labels could in theory also diminish overall prediction quality. For example, in the multi-space model every manifestation of stance results in inclusion of one additional modelling space. Therefore, given the fact that a number of additional parameter is introduced to the models for each additional space, the requirement on the size of the training dataset also increases proportionally. However, although this particular breaking news dataset is only of medium size, no positive effect of stance label exclusion could be noticed in the experiment. It is therefore reasonable to assume that all stance label manifestations provide information towards the classification target label. Furthermore, it has to be considered that excluding stance labels from the sequences not only eliminates the label but also effectively shortens the sequence. As has been seen throughout this section, the joint model can make use of each observation's posting time information. Exclusion of observations is therefore not advisable even when the associated stance label might contain little predictive value.

Specific requirements in experiments made it necessary to select samples of dataset $detection_{auto}$ to exclude the composition of the dataset from the experimental variables. As a side effect of repeatedly training the main classifier, a classification performance bias depending on considered sequences can be observed. For instance, the difference in F1-performance when using 1304 conversations compared to using all 4604 conversations is close to five percentage points for the multi-space model. However, performance is not simply a function of dataset size. This can be seen by the results of the third experiment where the same model performs worse when using 3513 conversations compared to the other two datasets. Obviously, some conversations are harder to classify correctly than others, a statement that can also be confirmed by comparing the predictions on conversation level between the CRF and the HMMs. Out of the 4604 conversations rumourous status could be determined successfully by both models in 1797 cases. In contrast, 431 conversations could not be classified correctly by either of the models.

Apparently, there is still work to be done before rumours can be detected automatically with high reliability. The role of HMM classifiers in this process could still be important, although they do not produce state-of-the-art results in terms of precision. Since also the currently best performing CRF produces wrong classification in more than one out of every three predictions, manual verification of rumourous status might not be avoidable in the near future when circumstances require very high precision. Hidden Markov classifiers are high recall systems and as such they can be used as a filter, drastically reducing workload before conversations' rumourous property is finally determined by a human expert. However, in the end CRFs and HMMs are fundamentally different in their respective strengths. Therefore, the overall question which system is best to use for the task comes down to how the cost of both error types are assessed.

Sarawagi and Cohen (2005) present an alternative approach to choosing one of the methods over the other. In their paper, the authors also confirm that HMMs as well as CRFs have their respective strengths and continue to propose a model called *semi-CRF* which is built for the named entity recognition task. Following their argumentation a similar joint approach could also be worth exploring for the rumour detection task in future work. Furthermore, apart from moving to related modelling techniques for the task, at least two additional aspects directly related to the HMM classifiers need to be evaluated for their impact on models' performance.

First, tweet stance in itself may not carry sufficient predictive value to be used as the only feature for the rumour detection task, even when modelled jointly with tweets' times. In fact, when the state-of-the-art CRF classifier is configured to perform classification solely on the stance feature its performance drops down tremendously to an F1-score of 0.422. Comparing this figure to the clearly su-

perior scores of the HMM classifiers demonstrates the HMMs' modelling power when information is limited. Nevertheless, it will be interesting to enhance the HMM classifiers by including additional features, as they have been employed successfully by Zubiaga et al. in context of CRFs. In principle, such extension can be performed in straightforward manner using the HMM framework developed in this thesis (Chapter 3). However, extraction and evaluation of feature candidates is out of the scope of this work and left for the future.

As a final consideration it is worth pointing out that the HMM-based classifiers naturally provide confidence values for their predictions. In principle these scores can be utilised to define thresholds which must be exceeded in order for a prediction to get accepted. If the criterion is not met, classification could then be relayed to an alternative classifier. The existence of confidence values is a systemic advantage of HMMs over the current state-of-the-art CRF since for the latter system confidence scores are not obtainable due to limitations in the software technology used. Therefore, HMMs are the more transparent method of providing classification labels.

Given conversation level predictions for the task at hand a simple classification pipeline composed of the HMM and the CRF also been briefly explored. For that purpose, final class prediction is determined based on the hidden Markov model's output for all instances where the confidence is higher than average confidence plus one standard derivation. All other instances' final class labels are determined by the CRF. Utilising both models' predictions in a simple pipeline already moderately increases performance to an F1-score of 0.610 and exploring a more sophisticated setup may prove to be worthwhile in the future..

5.4 Subtask II: Veracity Classification

After establishing that the HMM framework can be used to detect rumours on Twitter with high recall, it is now applied to the rumour veracity classification task, which is the final step in the pipeline proposed by Zubiaga et al.. Although this task is structurally related to the one discussed above, no assumptions on how the rumours were detected or tracked are made. In principle, any extraction method can be used before applying the framework to the veracity classification task as long as tweet level stance annotations are available. In the following sections it is detailed how two evaluations schemas were used to determine classification performance under varying conditions.

First, once more leave-one-event-out cross-validation is conducted to investigate the classifiers' overall performance on the five breaking news events, to determine their capability to perform ad-hoc veracity classification and to test their

robustness when noisy stance labels are used. Since the datasets $veracity_{gold}$ and $veracity_{auto}$ are substantially smaller than $detection_{auto}$, experiments on the individual stance labels' predictive power could not be repeated in context of veracity classification. However, in the rumour detection task best results were achieved when all stance labels are utilised in the models. Given the similarity of the tasks, these results are used as a proxy and all veracity classification evaluations make use of all four stance labels.

In order to appropriately assess the results of the leave-one-event-out cross-validation mainly two baselines are used. Both baselines were created in the scope of a research collaboration which also yielded a paper reporting many of the results presented below (Dungs et al., 2018). Due to their importance the baselines are also introduced in the following paragraphs.

To complement the leave-one-event-out cross-validation additionally the SemEval 2017 task 8 b closed setting is replicated. In this evaluation, the classifiers' performance is compared against the competition winners on a mixed evaluation dataset containing rumours from unseen as well as previously known events.

5.4.1 Stance Unaware Baseline B1

The first baseline *B1* performs veracity classification based on features commonly used in natural language processing tasks. As proposed by Castillo et al. (2011) and refined by Aker et al. (2017b) overall 33 features are utilised, including syntactical, semantic, indicator, user- and message-specific items. Despite the extensive feature set, this classifier remains oblivious of tweets' stance information. The actual classification in this setup is performed by using random forests, since they performed best out of a set of standard machine learning approaches tested on the feature set, for example, decision trees, k-nn and others.

5.4.2 Stance Aware Baseline B2

The second baseline *B2* extends on the first one by also including the stance information. More precisely, while keeping the feature set from B1, stance is included as additional features by following the method proposed by Liu et al. (2015a). In their work the authors include stance as a feature referred to as *relative stance score*, which is defined as the percentage of supporting (denying, questioning, commenting) tweets divided by the total number of tweets in a rumour. Since there are four different stances in total, four features are added to the classifier. Finally, once more random forests are used to perform the classification.

5.4.3 Overall Veracity Classification Scores

In this section the overall classification scores are reported based on the leave-one-event-out cross-validation performed on the rumour dataset $veracity_{gold}$. Table 5.6 details precision, recall and F1 scores as a weighted average across the five events for the baseline B1 and B2. Furthermore, scores are reported for the HMM-based classifier using stance as the only feature (θ) and the multi-space classifier θ', which uses stance and tweets' posting times as joint feature. For completeness, Table 5.6 also includes the score of using simplistic event level majority voting.

Table 5.6: Weighted average classification scores using dataset $veracity_{gold}$

System	Precision	Recall	F_1
B1	0.650	0.481	0.553
B2	0.661	0.481	0.557
θ	**0.747**	0.765	0.756*
θ'	0.690	**0.963**	**0.804***
majority-vote	0.059	0.025	0.035

* indicates significant difference to B1 and B2 (Tukey's HSD $p < 0.05$)

Overall, the F1-scores for the stance-unaware baselines B1 and the stance-aware version B2 are very similar (B1: 0.553 vs. B2: 0.557). This shows that there is only minimal benefit of naïvely including the stance information as an additional feature when using standard machine learning methods for veracity classification. When using HMM-based classifiers, however, already using stance as the only feature (θ) results in a far superior F1-score of 0.756. The performance is increased further to 0.804 for system θ' also including the tweets' times. Finally, it is also shown that usage of majority voting is an unsuited method to predict rumour veracity given this set of events. Overall, when using Tukey's honest significance test with $p < 0.05$, it can be established that both HMM-based classifiers θ and θ' perform significantly better than both baselines B1 and B2.

When considering precision and recall separately, again the HMM-based methods outperform both baselines. Looking at the recall alone, it can be noted that in particular the system θ' performs exceptionally well with a score of 0.963 while system θ still achieves a score of 0.765. Both baselines fail to recognise more than half of the true rumours, featuring an identical recall of 0.481. When regarding the precision scores it is noteworthy that system θ performs best (0.747) even outperforming system θ' which achieves a score of 0.690.

Table 5.7 gives an overview of all systems' performances on the level of individual events. Comparing the results, it can be seen that in three out of the five runs system θ' achieves the best scores (5.7a,5.7d,5.7e). System θ performs best when tested on the other two events (5.7b,5.7c). It is also striking that the performance varies substantially across the events and all classifiers. In the extreme case of the Ferguson riots event (5.7b) the baselines B1 and B2 fail completely. In contrast, system θ produces perfect veracity classification when being tested on rumours related to the Germanwings plane crash (5.7c). In the case of the events Ottawa shooting and Sydney siege (5.7d,5.7e) all systems achieve satisfactorily precise results, however, only the HMM-based classifiers also have a high recall leading to their superior F1-score on the data related to these events.

Investigating further on the conspicuously low F1-score of system θ' in case of the Ferguson riots event (5.7b) reveals that these particular result is likely to be caused by the imbalanced of the test sample. Since only two out of the 34 rumours are actually true, missing a single rumour has drastic impact on the test score. However, when considering the complete confusion matrix, it becomes apparent that the classifiers' performance on rumours regarding this event is closer as the F1-score suggests. For example, the distribution of true positive classification given the four methods is θ: 2, θ': 1, B1: 0, B2: 0 and the false positive classification distribution is θ: 1, θ': 3, B1: 4, B2: 5.

5.4.4 Performing Ad-Hoc Rumour Veracity Classification

Up to this point veracity classification results were obtained by using all available tweets in each rumourous sequence as classification basis. One of the long term goals of an automated rumour veracity classification system is its on time application to newly emerging rumours to produce a veracity label as soon as possible. If these labels can be produced reasonably fast while also maintaining satisfactory accuracy levels, the classifier can be used to support a timely flagging or removal of rumours on social media channels, potentially reducing their impact and reach in the community.

To investigate the ability of the HMM-based classifiers to aid in this process, ad-hoc classification was performed by considering only the first ten (first five) tweets in a sequence during test time, discarding any additional information contained in the sequence. In doing so, the classifier is challenged to perform based on the first few observations only comprising the rumours' early stance distribution patterns. The eventual resolution of the rumour is not included in the data fed to the models.

Table 5.8 details the F1-scores of both HMM-based classifiers when using the shortened sequences. It can be seen that for shorter sequences the classifiers'

Table 5.7: Performance comparison across all breaking news events

(a) Charlie Hebdo Attacks

System	Precision	Recall	F_1
B1	0.634	0.792	0.704
B2	**0.667**	0.667	0.667
θ	0.643	0.750	0.692
θ'	0.605	**0.958**	**0.742**

(b) Ferguson Riots

System	Precision	Recall	F_1
B1	0	0	0
B2	0	0	0
θ	**0.667**	**1**	**0.800**
θ'	0.333	0.500	0.400

(c) Germanwings Plane Crash

System	Precision	Recall	F_1
B1	0.333	0.500	0.400
B2	0.500	**1**	0.667
θ	**1**	**1**	**1**
θ'	0.500	**1**	0.667

(d) Ottawa Shootings

System	Precision	Recall	F_1
B1	0.818	0.450	0.581
B2	**0.909**	0.500	0.645
θ	0.882	0.750	0.811
θ'	0.792	**0.950**	**0.864**

(e) Sydney Siege

System	Precision	Recall	F_1
B1	0.714	0.303	0.426
B2	0.647	0.333	0.440
θ	**0.758**	0.758	0.758
θ'	0.750	**1**	**0.857**

performance drops down gradually. Compared to using the full sequences—which have a median length of 14 tweets—system θ shows an approximately linear degradation when shortening input length. However, test sequences of ten tweets still contain enough information for this classifier to outperform both baselines making use of all available tweets at test time. Solely shorting the input further to five tweets renders the result useless when considering the tweets' stance alone.

Table 5.8: F_1 scores for performing early classification

System	All tweets	First 10 tweets	First 5 tweets
θ	0.756	**0.658**	0.524
θ'	**0.804**	0.642	**0.618**

Moreover, looking at these additional scores, it becomes once more apparent that including the tweets' times benefits the classification process. Although system θ' also shows gradual performance drop when using short sequences, it is still able to outperform both baselines' best results with an F1-score of 0.618 while using only five tweets as input. This demonstrates that the joint modelling of stances and tweets' times is useful to perform timely veracity classification of rumours.

5.4.5 Using Automatically Generated Stance Labels

Considering the classifiers' perspective application in real time rumour veracity prediction, it is unlikely that human stance label annotations are widely available. Even when the associated cost of manual label creation is left discounted for, the process of manually creating annotations is also simply to slow. Therefore, the evaluation setup is again modified, this time performing leave-one-event-out cross-validation using the automatically generated stance labels contained in dataset $veracity_{auto}$. If the system is also able to accurately predict veracity labels using automatically created stance labels, it can also be applied to a large quantities of conversations in real time.

Table 5.9 details the results of an evaluation run using automatic stance labels for the HMM-based classifiers. Both systems show only marginal changes in their respective F1-scores compared to using gold stance labels. When considering precision and recall individually, it can be observed that system θ shows a decrease in precision from 0.747 to 0.632 when using automatic stance labels. Merely since recall improves with a similar magnitude the F1-score remains stable between the conditions. In contrast, model θ' is not impacted by using automatically generated labels even in terms of individual recall and precision scores. This again demonstrates the robustness of this particular classifier achieved by inclusion of the tweets' times in a joint feature.

Table 5.9: Classification scores using automatic stance labels

System	Precision	Recall	F_1
θ	0.632	0.888	0.738
θ'	**0.669**	**0.975**	**0.794**

5.4.6 SemEval 2017 Task 8B (closed) Revisited

The SemEval (short for Semantic Evaluation) series is an annual evaluation of computational semantic systems exploring the automated and formal analysis of meaning in language. In 2017 SemEval focussed particularly on three different aspects: Detecting sentiment, detecting humour and truth as well as the parsing of semantic structure. To this end, in total twelve tasks including suitable training data were created, inviting researchers to create and submit systems tailored to solving these tasks. After a fixed window of time the challenge is closed and all submitted systems are evaluated by the SemEval organisers based on the same test dataset which was not made available to the participants. Using the dataset introduced in Zubiaga et al. (2016), Task 8 specifically addressed rumour veracity and support. The task is further divided into two subtasks, where Subtask A covers stance label classification and subtask B deals with veracity classification. Finally, subtask B is divided into two variants: In the open variant participants could make use of additional resources (for example Wikipedia articles) for the classifications. In the closed variant all predictions had to be made solely based on the tweets' contents. The last setting exactly matches the setup as it is analysed in the previous sections, apart from the fact that the dataset provided in SemEval also contains the veracity label *unknown*.

The outcome of Task 8 A and B in open and closed variant are discussed in the work by Derczynski et al. (2017). Out of the 13 participants five provided solutions for subtask B. All five participating teams chose the closed variant of the subtask (Chen et al., 2017; Enayet and El-Beltagy, 2017; Singh et al., 2017; Srivastava et al., 2017; Wang et al., 2017). Singh et al. additionally provided a solution to the open variant. Enayet and El-Beltagy, Wang et al. and Singh et al. viewed veracity classification as a three-way problem, whereas Srivastava et al. and Chen et al. disregarded the rumours of unknown veracity creating a binary classification problem.

The challenge creators ranked participating systems' performances based on two measures. Classification accuracy is used as the primary quality criterion. Furthermore, all systems were required to provide confidence scores on rumour level for all veracity predictions. Given a pre-defined reference confidence score, each system's final performance is calculated by normalising respective accuracy scores

using the confidence root mean square error (RMSE). The entire evaluation procedure was also made available as a Python script[2] after the challenge was closed.

According to the evaluation results, the systems IKM (Chen et al., 2017) and NileTMRG (Enayet and El-Beltagy, 2017) tie for the best performing classifier with an accuracy score of 0.536. It is interesting to note that system IKM employes a binary view on the problem while NileTMRG performs three-way classification. The latter is also the only system that makes use of the stance feature. However, it is still mainly performing conventional natural language processing and makes use of stance only in terms of simple percentage scores. The other systems' performance falls short of the winnings systems' with final scores ranging from 0.286 to 0.464.

To test the HMM-based classifiers' performance in light of the competing systems, the best performing system θ' using stance and time as a joint feature has also been evaluated following the SemEval procedure. Only minor modifications were made in that context. First, the classifier was extended to include confidence scores. The general class assignment formula (Equation 5.1) considers sequences probabilities under condition of each class specific model. These calculations are conducted in log-space to avoid arithmetic problems with particularly small probability values. As proposed by Vosoughi (2015), the confidence of two probabilities in logarithmic space can be estimated by calculating the absolute value of the subtraction of both values. However, since sequences' probabilities are eminently influenced by sequence length and this procedure does not include a normalisation step, the calculated confidence scores show substantial variance.

Since disclosure of *normalised* scores was one of the SemEval requirements, instead the softmax function (Bishop, 2006, Page 198) is used for calculating confidence values. Apart from score normalisation, the function has the further advantage that it can be applied to multi-class problems as well. Therefore, in the following, confidence of assigning a sequence ε_i to any class c out of the possible class values C is calculated using the formula detailed in Equation 5.3.

$$Confidence(\varepsilon_i \in c) = \frac{e^{\mathbb{P}(\varepsilon_i|\theta_c)}}{\sum\limits_{c \in C} e^{\mathbb{P}(\varepsilon_i|\theta_c)}}, \text{ for } C = \{true, false, \ldots\} \tag{5.3}$$

It is worth pointing out that for very small classification probabilities a default confidence score of 0.5 is assigned for the binary classification problem to avoid arithmetic underflows. Nevertheless, the class value is still predicted following Equation 5.1.

After modifying the framework, the model has been retrained on the same data

[2] `http://alt.qcri.org/semeval2017/task8/data/uploads/scorer.zip`—accessed 24.10.2018

provided to the challenge participants. Overall the data comprised eight events—including the five previously discussed breaking news events—and a total of 297 conversations. Out of those 23 originate from the three new events. Furthermore, 25 conversations in the training set are marked as dedicated validation examples. Hence, these sequences were used to tune the model's parameters and to select the best performing model configuration. Finally, performance of system θ' was evaluated based on 28 test rumours using the evaluation script as provided by the challenge organisers.

Table 5.10 gives an overview of the performance of the original challenge participants as well as the HMM-based classifier θ'. It can be seen that the HMM is able to tie the accuracy scores of the challenge winners, while showing lower confidence root mean square error (0.571) compared to the other systems (RMSE 0.672 and 0.763). It is worth noting that the evaluation procedure effectively punishes the binary classification systems IKM, DFKI-DKT and θ' by regarding all eight test sequences with *unknown* veracity value as misclassified. Nevertheless, both best performing binary classifiers are able to match the score achieved by the three-way classification system NileTMRG.

Table 5.10: Comparison of performance in the SemEval Task 8B (closed) challenge

System	Accuracy	Confidence RMSE
NileTMRG	**0.536**	0.672
ECNU	0.464	0.736
IITP	0.286	0.807
IKM	**0.536**	0.763
DFKI DKT	0.393	0.845
HMM θ'	**0.536**	**0.571**

As the final experiment, the possibility of extending θ' to model veracity as a three-way classification problem was also briefly explored. Therefore, a third multi-space HMM was created and trained on the $veracity = unknown$ condition using the training data provided. Additionally, test sequence veracity is predicted using a variant of formula 5.1 considering the three class values *true*, *false* and *unknown*. When using the SemEval evaluation procedure, the multi-class version of system θ' achieves an accuracy score of 0.214 and a confidence root mean square error of 0.895, clearly falling short of the challenge competitors' performance.

In the following section, the results of the various experiments are discussed, investigating the overall suitability of HMM-based methods for veracity classification. Furthermore, a number of potential improvements especially regarding

the unsatisfactory multi-class performance as well as further research directions are outlined.

5.4.7 Discussion and Future Work

Looking at the individual experimental results detailed above, it can be summarised that two variants of HMM-based classifiers can successfully be used for the veracity classification task. Both principal variants make use of tweet stance as the primary feature and are outperforming stance unaware baseline approaches using natural language processing. Even when a simple stance count-based feature is included in baseline B2, it can not match the performance of the HMM classifiers.

Given the detailed performance figures, it can also be seen that inclusion of tweets' times together with stance as a joint feature greatly benefits the classification process. For example, θ' correctly classifies twelve rumours where all other classifiers fail—which only happens in one and two instances for the baselines B1 and B2 respectively. Moreover, nine out of the twelve correctly classified rumours are in fact true while they feature only one to three supporting stance observations. Apparently, high level collective stance patterns are indicating rumour veracity, as it is for example also described in the work by Mendoza et al. (2010). Given the results obtained here, these patterns can best be exploited by using the joint feature of tweets' stance and time. In contrast, simple stance counting is insufficient as can be seen by the poor performance of system B2.

Overall, the multi-space HMM makes use of this joint feature to achieve the best performance compared to self-created baselines. Moreover, the model also ties the scores of the best performing models amongst a number of competing systems. These are state-of-the-art deep neural approaches, that were recently developed in the scope of the SemEval veracity classification challenge. Therefore, it becomes apparent that hidden Markov models are not only versatile, but also remain a relevant tool despite the technology's age.

The additional experiments on early rumour classification as well as on the usage of automatic stance labels both strengthen the argumentation towards preferring the multi-space model θ' for the task. On the one hand, the model is able to maintain its classification performance while using automatically generated stance labels. Even though such automatic generation inevitably introduces a certain degree of noise to the system, stable performance of model θ' indicates advancement in overcoming the need for manual annotations on tweet level for completing the task. On the other hand, θ' is also able to outperform the baselines even when using only the first five tweets of a conversation as a classification basis. Naturally, the performance of θ' drops when reducing sequences' length, since

collective stance is expected to stabilise only over time. However, especially when also considering that the stance-only model θ shows higher sensitivity to noise and has faster deteriorating performance on short sequences, this again shows the benefit of the joint modelling of stance and time. Only the elaborate multi-space version is able to accomplish this joint modelling, justifying the higher modelling effort compared to discrete models. Additionally, also the discrete model, whilst being restricted to using only the first ten stance observations, is able to outperform both baselines using all available information. This result continues to highlight the overall importance of stance as a feature to capture user behaviour in the veracity classification task.

For practical application of the veracity classifiers their ability to generalise to new and unseen events will be the most crucial success factor. As it is also established by Zubiaga et al. (2016), the breaking news events all have specific particularities in terms of their rumours' lifecycle timelines and thus are attracting different levels of user attention of varying longevity. This results in event specific activity patterns concerning users' (re-)tweeting behaviour as well as regarding the prevalence of the four stances in their responses. Accurately capturing these developments is a challenge especially for the HMM-based classifiers as they strive to map the tweets' stance sequences to the conversations' veracity value. For generating a high quality classifier, special care needs to be taken to avoid overfitting the training data, ensuring that the system remains general enough to adapt to new arising rumours in previously unseen events.

As a consequence, all but the last experiment replicating the SemEval setting use the leave-one-event-out evaluation procedure to precisely address this generalisation problem. Compared to, for example, ten-fold random sampling, holding an entire event for evaluation purposes has the advantage of approximating a classifier's real application as closely as possible. Other evaluation techniques could be used to achieve better scores on the available datasets. However, the results would be biased in a sense that the classifier is then trained on event specific patterns that are not known for new data. Therefore, using leave-one-out-evaluation is preferable in this context as it gives more reliable results.

When considering the five breaking news events, the impact of the events' specifics is highlighted by all classifiers' large performance variance across these events (Table 5.7). For example, F1-scores of system θ' are ranging from 0.400 for event Ferguson riots to 0.864 for event Ottawa shootings. Large differences can also be noticed regarding recall and precision, even when excluding outlier results. For example, the precision of system B1 ranges from 0.303 for event Sydney siege to 0.792 in case of the Charlie Hebdo attacks. Additionally, limitations of the dataset also have to be emphasised. Especially the events Ferguson riots and Germanwings plane crash are unbalanced in their veracity value sparking far more false than true rumours. Furthermore, only twelve rumours were sampled

from the latter event (see Table 5.2). Both facts could also partially account for the variance in classification performance, particularly since F1-scores are used as the main evaluation criterion which do not take true negative classifications into account. F1-scores—as well as accuracy used in the SemEval series—have the advantage of being simple and intuitive. Therefore, they are also widely used to evaluate classification systems. However, Powers (2011) argue that classifiers should instead be evaluated based on more sophisticated unbiased measures to avoid false assessment of systems' performance.

In future work using such measures for evaluation could be investigated. Especially in context of the veracity classification task, it is not necessarily more desirable to tune the classifier to accurately identify true rumours over false rumours, since none of the class values is more important than the other. In other applications, however, there clearly is an actual target class value whose correct identification motivated creation of the classifier, for example, correctly identifying a rumour in the rumour detection task. In the latter cases the F1-measure might still be the appropriate choice. Furthermore, it is also worth exploring in future work, how advanced measures as proposed by Powers could be used during model training phase to select the truly best performing models out of the large number of combinations of model size and initial parameter configurations.

Apart from exploring different evaluation criteria, there is further room for improvements in future work, especially when considering veracity classification as a three-way problem. Inspired by the SemEval Task 8B setting, the best performing HMM system θ' was extended to explicitly model unknown veracity as well. Using a rather straightforward way of training a third model on the new condition, class prediction is made solely based on the highest sequence probability score given by Equation 5.1. As a result, performance of the multi-class version of θ' fell short of the competition by a large margin. It might be argued that inclusion of a third model disproportionally increases model parameter count, thus complicating parameter tuning given the limited training data for the task. However, all systems competing in SemEval have been trained using the same amount of data. Therefore, it is evident that three-way classification has to be tackled using more sophisticated means when using HMM-based classifiers. Lukasik et al. (2015) propose a one-vs-all classification approach for all labels, finally assigning the label that overall has the highest probability. However, implementation of this method would require to retrain all models for the one-vs-all setting and is left for future work.

5.5 Transparency of the Classifiers

Transparency is an important aspect of every machine learning application, which needs to be addressed accordingly. However, not least caused by the recent advance of deep learning approaches, it is often neglected in practice. Yet, especially given the delicacy of (falsely) flagging a conversation as rumourous, any rumour classifier should ideally provide justification for its decisions. In productive systems deciding classification factors could also be communicated to the users, potentially increasing acceptance and thus the impact and benefit of the system.

In this regard, HMM-based classifiers are superior compared to deep learning approaches (these were prevalent also in the SemEval 2017 challenge), since HMMs make direct use of the original features which relate to the real world phenomenon—in this case the users' conversation. While conclusively determining the reason for a classification in the individual case is certainly a challenge, differentiated investigation of tuned models' parameters could at least constitute an opportunity to discover general tendencies. The considerations given in this section should therefore be regarded as a first step towards ultimately creating a transparent rumour classification system.

Due to the large number of individual experiments that were conducted in both rumour classification tasks as well as the cross-validation procedure, models can only be discussed exemplary here. Focussing on the best performing model configurations created in the most general experiments, two discrete and two multi-space models featuring at most four hidden states were randomly sampled for further discussion.

The first discussed set of models using configuration θ is sampled from one of the cross-validation runs in the rumour detection task using dataset $detection_{auto}$. Figure 5.8 details the model trained on conditions *rumour* (5.8a) and non-rumour (5.8b)[3]. It can be seen that training the models on the two conversation sets has a distinct effect on the models' final properties, particularly on the starting state (double circle). In both models a single state has 1.0 probability of being the starting state, however, this is s_2 in the *rumour* model and s_1 in the *non-rumour* model, when models are arranged to maximise structural similarity.

Despite their similar structure, the models' hidden state transition matrices as well as their emission matrices show differences. Comparing the parameters across the conditions, it can be seen that in the non-rumour condition a supporting stance is likely to be observed at the beginning of a sequence (the first observation to be precise). Since in this model s_1 is the only state which is featuring a high probability of emitting Supporting, while at the same time it is almost never

[3]Note that the sum of the states' transitions or emissions may not add up exactly to 1.0 due to rounding.

reached once it is left (i.e. transitions $s_2 \rightarrow s_1$ and $s_3 \rightarrow s_1$ are unlikely to occur), a supporting observation becomes very unlikely later in the sequence. The model trained under rumour condition is contrasting this pattern. In this model state s_1 has a 84% probability of emitting Supporting. However, looking at the state transition matrix, it can be established that this state is also more likely to be entered at a later point in the sequence—if at all.

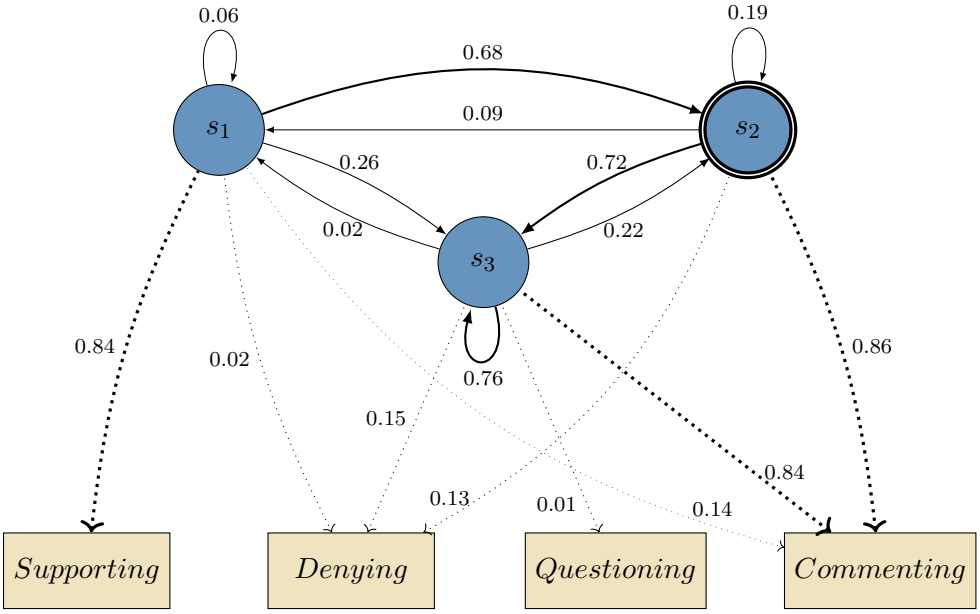
Certainly, investigating a single model pair does not generate sufficient evidence to derive and support a classification rationale that can be communicated to end users. However, in principle similar patterns could also be present in the other models generated for the rumour detection task and in that case generalisation becomes more reasonable.

Investigation of the second set of models, two four state discrete HMMs used for rumour veracity classification (Figure 5.9[4]), reveals another interesting pattern. In this instance, the hidden process (Figure 5.9a) of the model trained on the *true* condition will eventually reach state s_3, which has a 92% probability of looping while emitting Commenting in 82% of the cases. Since a similar constellation can not be observed in the opposing model trained on the *false* condition (Figure 5.9b), it appears that longer sequences of neutral responses in the later stages of a rumour could be indicative of the rumour's truthfulness. As already discussed with respect to the rumour detection model, this pattern needs to be confirmed by other models created in the same experimental setting. In case a generalisation can be established, it is a first step towards creating a thorough understanding of the classifier.

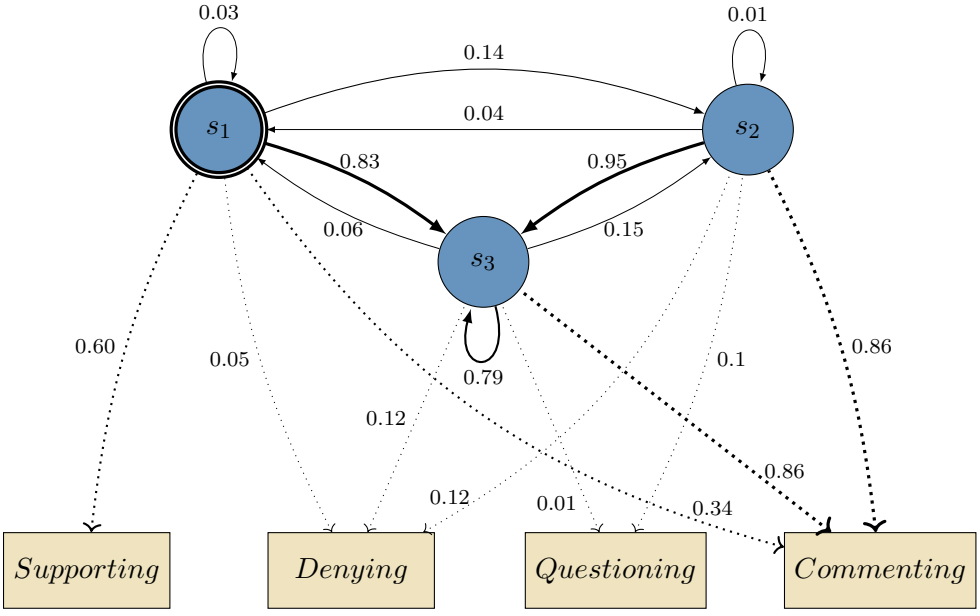
So far two examples of discrete HMMs were used for illustration purposes although these models were outperformed by the multi-space modelling variant θ' in both main experiments. However, as (multi-space) model complexity grows with increased model performance, simultaneously any potentially inferable patterns become more and more opaque to intellectual analysis. Even when confining investigation to smaller multi-space models with two to four hidden states, it is a challenge to make direct use of the models' probability density functions when striving to achieve transparent classification.

To illustrate this difficulty, a two hidden state multi-space veracity classification system is used. Both models' multi-space emission functions' rate parameter—i.e. the slope of the spaces' function graphs in Figure 5.7—can now be compared on two levels: (1) With respect to the other state in the same model and (2) with respect to the other model's hidden states. In doing so, it can be noted that overall the rate parameters differ only marginally, both across models as well as across the states in an individual model. The largest difference in the emission

[4]Both models have $> 99\%$ probability of starting in state s_1. For easier visual comprehension the models' emissions are omitted from the figure.

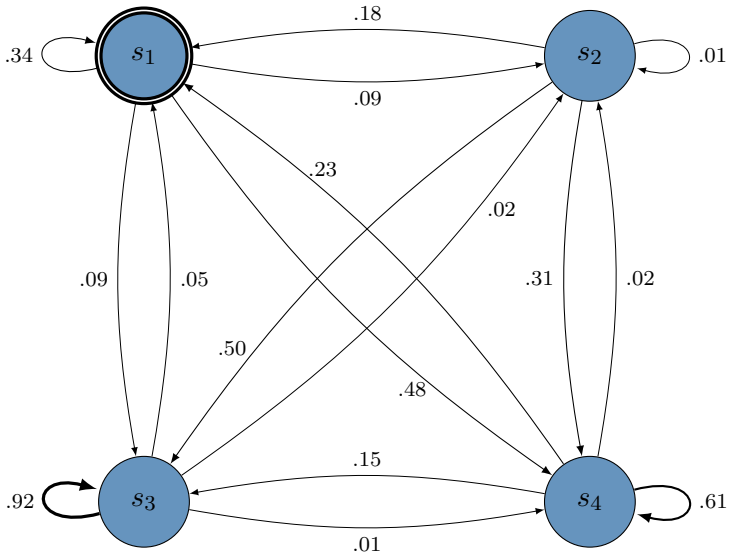


(a) Model trained on *rumour* condition

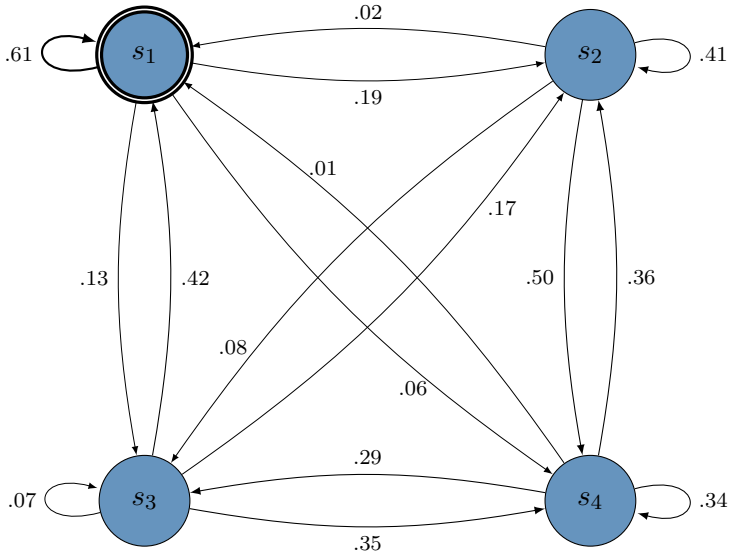


(b) Model trained on *non-rumour* condition

Figure 5.8: Discrete models used in the rumour detection task



(a) Model trained on *veracity true* condition



(b) Model trained on *veracity false* condition

Figure 5.9: Hidden process of models used in the veracity classification task

functions' rate can in fact be observed for the spaces Denying and Questioning between the two hidden states of the model trained under condition *true*. Based on this, it can be assumed that true rumours can be characterised by at least two temporal patterns of stance distributions which are reflected accordingly in the hidden states properties. However, in all other cases pairwise comparison does not reveal noticeable differences.

Since the multi-space models' power was undoubtedly shown in the experiments discussed above, it has to be noted that the basis of classification decision-making obviously lies hidden inside the models overall parameters. In this case, a conclusive analytic deduction of the classifiers' decisions remains a task for future work. Nevertheless, especially in comparison to deep learning approaches HMMs are more transparent despite the obvious challenges.

At least for the discrete models it was also shown how these models can be exploited in principle to achieve transparent classification. Therefore, a reasonable practicable approach might be to use a combination of discrete and multi-space models to build a composite classifier. Classification rationale can then be generated based on the discrete modelling variant, at least for instances where models are in agreement.

5.6 Summary of HMM-Based Rumour Analysis

In the previous chapter it was demonstrated how the HMM framework can be utilised to build classification systems which are used for the analysis of potentially rumourous conversations on Twitter. As it has been established in related work, the prospective goal of establishing automatic rumour veracity classification has high topicality. A large corpus of related work on this problem exits, often involving deep neural approaches typically based on features generated through natural language processing. Based on these developments, substantial improvements in terms of overall system performance could be achieved in the recent past.

Complementing the predominantly used techniques the HMM framework features an alternative view on veracity classification by using tweets' stance as the main feature. Especially by joint inclusion of tweets' posting times in the stance feature the Markov models make strong use of the sequential characteristics of the users' conversations to model the rumourous property. The rumour detection task naturally stands before the former task as part of a larger pipeline. It is structurally related to veracity classification and can also be viewed as a binary classification problem. The similar characteristics of the two tasks allow the framework to be applied as a classifier in both cases with only minimal modifications.

Applying the HMM framework to the problems provided valuable contributions to the field which can be exploited in future systems. Since veracity classification can be performed by using stance and tweets' times at the same quality level as is achieved by feature rich deep neural approaches, the power of the collective stance feature could clearly be demonstrated. Given that there also has been advancement in creating automatic stance classifiers, the HMM framework provides a mean to abstract from textual content for the veracity classification task. Even when considering that stance extraction is—at least for the time being—based on the tweets' content, the HMM classifier is still less feature heavy than related work. In contrast, especially deep neural approaches also make heavy use of social features which are more costly to acquire at larger scale. However, it has to be noted that for the rumour detection task the feature rich natural language processing approach did produce the better results. Yet, in terms of recall the HMMs could really shine providing scores in the 90% range and above.

Both applications of the framework also strikingly demonstrate how the theory of multi-space HMMs can be applied to domains unrelated to the ones it originally was designed for. The main strength of this approach is the unification of discrete and continuous signals into a single model. Looking at the results presented above, it is clearly demonstrated that the modelling process indeed benefits from the enhanced expressiveness of the models, as the multi-space variant of the classifier performs better under almost all conditions. However, also the discrete models should not be dismissed completely as they have the advantage of simplicity and can also prove to be an asset when priority is given to creating transparent classifiers. Since the in comparison more elaborate multi-space models quickly become very hard to interpret visually and intellectually, the latter especially shine when sheer classification performance is favoured.

The applicability of the HMM framework as a classifier on a general level and the usefulness of the stance feature particularly for the veracity classification task was demonstrated above. However, at its current state this work also leaves challenges for future work.

On the one hand, some of the experiments made it necessary to exclude a number of conversations from the dataset in order to isolate the targeted independent variable. In other cases conversations had to be excluded because of missing ground truth information on the class label. Consequently, some of the experiments were conducted on smaller than ideal sample size—especially the manually curated dataset $veracity_{gold}$ only contains 197 conversations. Moreover, particularly in the rumour detection task it became obvious that the dataset sample can have strong influence on classification scores. Both considerations raise the question of generalisability of the results, although leave-one-event-out cross-validation showed that the models in principle work well with previously unseen events. However, it will be interesting to observe how the methods presented here per-

form when they are being tested on completely unrelated events, possible being extracted using other than Zubiaga et al.'s methodology.

On the other hand, the problem of appropriately selecting features still exits in hidden Markov modelling approaches. Since this technique does not feature inherent automated selection methods, special attention has to be given to the question which features to best include for which task. In the scope of this work, a small feature set was preferred to demonstrate the overall applicability of the presented methods. However, stance may not be sufficient to model the rumourous property alone, as the respective models fail to constitute the state-of-the-art on the rumour detection task. This circumstance might be caused by the fact that automatically generated stance labels were used for the task. As it was not tested on this type of data, the stance classifiers' performance might be unsatisfactory when applied to non-rumourous conversations. Independently whether this it true or not, the HMM-based rumour detection is improvable in terms of precision and might benefit from the addition of advanced features to better catch the users' responses.

Furthermore, it might simply be insufficient to model the rumourous status as a binary classification problem. Instead it could be more appropriate to use a continuous scale, reducing false classification rate. It remains a question for future work how, for example, the confidence values generated by the HMM framework can be used to accomplish this. In this context, it is again worth mentioning that the state-of-the-art rumour detection CRF does not feature any confidence scores which puts the HMM framework in the advantage in that regard.

CHAPTER 6

Summary and Future Work

In this thesis user behaviour was quantitatively modelled as ordered sequences of actions, taking particular interest in uncovering hidden temporal patterns. Hidden Markov models are inherently well suited for tasks involving hidden components and provide an adequate balance between modelling power and complexity. While HMMs have previously been used extensively and successfully in many domains, common instantiations of this technique are limited by a conceptual drawback: Their inability to jointly model features on continuous and discrete scale. However, this is an essential prerequisite for satisfactorily capturing the temporal components of hidden patterns in user behaviour.

In this work, it was shown that this gap can principally be addressed in two ways: On the one hand, the theoretical particularities of HMMs allow domain knowledge to be included in the modelling process by purposefully initialising the models' structure on a semantic level. This allows to associate specific states with selected subsets of observations, efficaciously representing the discrete part of the interaction. Consequently, the temporal component can be modelled using conventional continuous emission functions. However, for this method to be applicable prior research on the subject is mandatory, which might not always be available in sufficient detail.

On the other hand, multi-space HMMs provide the extend mathematical background to successfully build joint models of user behaviour in domains where no prior knowledge is available. While this concept greatly increases the range

of problems that can be addressed with HMMs, multi-space models have so far rarely been applied in practice outside the domain of natural speech processing. As technical contribution of this work, a HMM framework was developed that can be used for creating conventional HMMs as well as multi-space models. Hereby, special care was taken of designing multi-space emission functions that are appropriate for modelling user action durations.

The applicability of the HMM framework was demonstrated based on two practical examples: First, mental states and reasoning effort of users engaged in interactive information retrieval was modelled. In this application, HMMs are used to complement prior qualitative work on the subject by providing a quantitative analysis of the retrieval process. Derived from the prior works, search sessions are divided in two phases: Searching, for which it is assumed that users start to familiarise themselves with the topic, and additionally the second phase Finding, which is characterised by activity patterns where effectiveness and efficiency is expected to increase.

While users' search behaviour has been studied extensively in the past, this work contributes to the field by merging a quantitative modelling approach with the search phase hypothesis formulated in qualitative information seeking models. Based on HMMs trained on search engine transaction log data, these prior models could be confirmed in a novel manner, showing that users find more relevant documents in the second phase of a search session while also taking less time for finding each relevant item compared to the earlier phase of the search.

The qualitative user model can also be utilised to uncover implicit information hidden in an unlabelled transaction log. For example, an additional analysis of the most likely hidden state path of each search session revealed that only approximately 37% of the search sessions reach the second phase. Consequently, the remaining searches are likely to be rather simple lookup tasks, which can be completed quickly and without considerable reasoning effort.

The joint modelling of users' search actions together with the actions' durations is also an important step towards creating accurate predictions of future developments in a complex search session. Predictions of users' search times can ultimately be useful for implementing search systems that are capable of providing sophisticated user guidance, for example, by following the interactive probability ranking principle.

Although a number of promising results could be achieved by describing search phases with HMMs, there is also potential for improvement. Certainly, a two-phase model of search is an oversimplification of reality and additional phases should be considered in the future. Additionally, in follow-up studies it will be interesting to create multi-phase search models, which allow the addition of further features to capture the user behaviour more precisely. Furthermore, the

models could be enhanced by allowing users to return to previous search phases, which more closely matches the dynamic of the users' information needs. In future work, models could also be adapted to a specific task type, which is likely to have substantial impact on the prevalence of interaction patterns in a search session. Lastly, fully personalised HMMs could be used in adaptive retrieval systems to create user specific recommendations.

The second application of the framework is contrasting the first in terms of subject and modelling outcome, which also highlights the framework's flexibility. Based on a set of potentially rumourous conversation held on Twitter, conventional and multi-space HMMs were used to build classifiers to determine the conversations rumourous property as well as the veracity of the detected rumours. In these two subtasks the HMM framework was utilised to address two of the four essential steps in automated rumours veracity classification, which is one of the high priority challenges in social media research of today.

The particular solution to these problems presented here is also contrasting prior work on the subject. Especially, by using tweet stance and time as the only features, this work provides an interesting alternative to common feature heavy and deep approaches. Additionally, this work does not only illustrate an alternative modelling of the rumourous property, it also achieves state-of-the-art results in the veracity classification task with an F1-score of over 80%. The same modelling approach also manages to outperform all participating systems of the SemEval 2017 rumour veracity classification challenge.

In some of the numerous follow up experiments, it was also shown that especially the multi-space HMM classifiers are robust to noise in the stance label. Additionally, this system is also able to perform veracity classification with reasonable performance after observing only five tweets, which continues to highlight the benefit of using multi-space probability distributions to realise a joint modelling of users' actions and the respective durations. While this work is mainly based on manually labelled tweets, an extension to automatically generated labels was also shown. In follow up work it will also be interesting to investigate the models performance on other social media data.

Classification performance in the rumour detection use case was not quite as convincing as for veracity classification, falling slightly short of the state-of-the-art CRF approach. However, the HMM classifiers still provide an interesting complement to the CRF by showing vastly superior recall. Certainly, successfully detecting rumourous conversations is a central factor with respect to the rumour classification pipeline's practical benefit, since initially missed rumours can spread without hindrance. In future work, it could therefore be worth exploring how both modelling techniques' strengths can be combined to build a unified system. Furthermore, precision of the HMMs in the rumour detection task could

also be improved by considering additional features, fully utilising the modelling capabilities arising from the advanced mathematics of multi-space probability distributions. Besides, ultimately the view on the classification problems needs to be broadened to include transparency aspects as well, which could prove to be another opportunity to take advantage of the functional principle of HMMs.

Part III

Appendix and Listings

APPENDIX A

Mapping of sowiport Log Entry Types

This table details the mapping of sowiport log entries to the four prototypical user actions (Query, Snippet, Abstract, Mark) applied in the search phase experiment. Entries unrelated to the actual search progress remain unmapped. Furthermore, the original log contains some redundancy which is also eliminated.

Table A.1: Mapping of sowiport log entries to user actions

Log entry	User action
`delete_comment`	—
`export_bib`	Mark
`export_cite`	Mark
`export_mail`	Mark
`export_search_mail`	Mark
`goto_about`	—
`goto_advanced_search`	Query
`goto_advanced_search_reconf`	Query
`goto_contribute`	—
`goto_create_account`	—
`goto_delete_account`	—
`goto_edit_password`	—

Log entry	User action
goto_favorites	Mark
goto_fulltext	Mark
goto_google_books	Mark
goto_google_scholar	Mark
goto_history	—
goto_home	—
goto_impressum	—
goto_last_search	Query
goto_Local_Availability	—
goto_login	—
goto_partner	—
goto_sofis	—
goto_team	—
goto_thesaurus	—
goto_topic-feeds	—
goto_topic-research	—
goto_topic-research-unique	—
purge_history	—
save_search	—
save_search_history	—
save_to_multiple_favorites	Mark
search	Query
search_advanced	Query
search_change_facets	Query
search_change_nohts	Query
search_change_nohts_2	Query
search_change_only_fulltext	Query
search_change_only_fulltext_2	Query
search_change_paging	Query
search_change_sorting	Query
search_from_history	Query
search_institution	Query
search_keyword	Query
search_person	Query

Log entry	User action
search_thesaurus	Query
search_with_CTS_possiblity	Query
select_from_CTS	Query
to_favorites	Mark
view_citation	—
view_comment	—
view_description	—
view_record	—
view_references	—

List of Figures

List of Tables

Bibliography

Ageev, M., Guo, Q., Lagun, D., and Agichtein, E. (2011). Find It If You Can: A Game for Modeling Different Types of Web Search Success Using Interaction Data. In *Proceedings of the 34th ACM SIGIR*, pages 345–354.

Aker, A., Derczynski, L., and Bontcheva, K. (2017a). Simple Open Stance Classification for Rumour Analysis. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 31–39.

Aker, A., Zubiaga, A., Bontcheva, K., Kolliakou, A., Procter, R., and Liakata, M. (2017b). Stance Classification in Out-of-Domain Rumours: A Case Study Around Mental Health Disorders. In *International Conference on Social Informatics*, pages 53–64. Springer.

Asmussen, S. (2003). *Applied Probability and Queues*. Applications of Mathematics : Stochastic Modelling and Applied Probability. Springer.

Baker, J. (1975). The DRAGON System—An Overview. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):24–29.

Bakis, R. (1976). Continuous Speech Recognition via Centisecond Acoustic States. *The Journal of the Acoustical Society of America*, 59(S1):S97.

Baum, L. E. and Eagon, J. A. (1967). An Inequality with Applications to Statistical Estimation for Probabilistic Functions of Markov Processes and to a Model for Ecology. *Bulletin of the American Mathematical Society*, 73(3):360–363.

Baum, L. E. and Petrie, T. (1966). Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *The Annals of Mathematical Statistics*, 37(6):1554–1563.

Baum, L. E., Petrie, T., Soules, G., and Weiss, N. (1970). A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *The Annals of Mathematical Statistics*, 41(1):164–171.

Baum, L. E. and Sell, G. R. (1968). Growth Transformations for Functions on Manifolds. *Pacific Journal of Mathematics*, 27(2):211–227.

Belkin, N. J. (1980). Anomalous States of Knowledge as a Basis for Information Retrieval. *Canadian Journal of Information Science*, 5(1):133–143.

Bellman, R. (1954). The Theory of Dynamic Programming. *Bulletin of the American Mathematical Society*, 60(6):503–515.

Bellman, R. (1957). A Markovian Decision Process. *Journal of Mathematics and Mechanics*, 6(5):679–684.

Bernardo, J., Bayarri, M., Berger, J., Dawid, A., Heckerman, D., Smith, A., and West, M. (2007). Generative or Discriminative? Getting the Best of Both Worlds. *Bayesian Statistics*, 8(3):3–24.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

Borisov, A., Markov, I., de Rijke, M., and Serdyukov, P. (2016). A Neural Click Model for Web Search. In *Proceedings of the 25th International Conference on World Wide Web*, pages 531–541. International World Wide Web Conferences Steering Committee.

Brin, S. and Page, L. (1998). The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117.

Cao, H., Jiang, D., Pei, J., Chen, E., and Li, H. (2009). Towards Context-Aware Search by Learning A Very Large Variable Length Hidden Markov Model from Search Logs. In *Proceedings of the 18th International Conference on World Wide Web*, pages 191–200. ACM.

Capes, T., Coles, P., Conkie, A., Golipour, L., Hadjitarkhani, A., Hu, Q., Huddleston, N., Hunt, M., Li, J., Neeracher, M., Prahallad, K., Raitio, T., Rasipuram, R., Townsend, G., Williamson, B., Winarsky, D., Wu, Z., and Zhang, H. (2017). Siri On-Device Deep Learning-Guided Unit Selection Text-to-Speech System. In *Proceedings of Interspeech 2017*, pages 4011–4015.

Cassandra, A. R. (1998). A Survey of POMDP Applications. In *Working Notes of AAAI 1998 Fall Symposium on Planning with Partially Observable Markov Decision Processes*, volume 1724.

Castillo, C., Mendoza, M., and Poblete, B. (2011). Information Credibility on Twitter. In *Proceedings of the 20th International Conference on World Wide Web*, pages 675–684. ACM.

Chen, H. and Cooper, M. D. (2002). Stochastic Modeling of Usage Patterns in a Web-Based Information System. *Journal of the American Society for Information Science and Technology*, 53(7):536–548.

Chen, Y.-C., Liu, Z.-Y., and Kao, H.-Y. (2017). IKM at SemEval-2017 Task 8: Convolutional Neural Networks for Stance Detection and Rumor Verification. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, pages 465–469.

Derczynski, L., Bontcheva, K., Liakata, M., Procter, R., Hoi, G. W. S., and Zubiaga, A. (2017). SemEval-2017 Task 8: RumourEval: Determining Rumour Veracity and Support for Rumours. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, pages 69–76.

Dietterich, T. G. (2002). Machine Learning for Sequential Data: A Review. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition*, pages 15–30. Springer.

Dungs, S., Aker, A., Fuhr, N., and Bontcheva, K. (2018). Can Rumour Stance Alone Predict Veracity? In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3360–3370. Association for Computational Linguistics.

Dungs, S. and Fuhr, N. (2017). Advanced Hidden Markov Models for Recognizing Search Phases. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval*, pages 257–260. ACM.

Ellis, D. (1989). A Behavioral Approach to Information Retrieval System Design. *Journal of Documentation*, 45(3):171–212.

Enayet, O. and El-Beltagy, S. R. (2017). NileTMRG at SemEval-2017 Task 8: Determining Rumour and Veracity Support for Rumours on Twitter. In *In Proceedings of the 11th International Workshop on Semantic Evaluation*, pages 470–474. ACL.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., and Thrun, S. (2017). Dermatologist-Level Classification of Skin Cancer With Deep Neural Networks. *Nature*, 542(7639):115.

Feng, S., Manmatha, R., and McCallum, A. (2006). Exploring the Use of Conditional Random Field Models and HMMs for Historical Handwritten Document Recognition. In *Second International Conference on Document Image Analysis for Libraries*, pages 1–8.

Forney, G. D. (1973). The Viterbi Algorithm. *Proceedings of the IEEE*, 61(3):268–278.

Fuhr, N. (2008). A Probability Ranking Principle for Interactive Information Retrieval. *Information Retrieval*, 11(3):251–265.

Goyal, A., Metallinou, A., and Matsoukas, S. (2018). Fast and Scalable Expansion of Natural Language Understanding Functionality for Intelligent Agents. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 145–152.

Hamidian, S. and Diab, M. T. (2015). Rumor Detection and Classification for Twitter Data. In *Proceedings of the 5th International Conference on Social Media Technologies, Communication, and Informatics*, pages 71–77.

Hamidian, S. and Diab, M. T. (2016). Rumor Identification and Belief Investigation on Twitter. In *Proceedings of the 7th Workshop on Computational Approaches to Subjectivity*, pages 3–8.

Han, S., Yue, Z., and He, D. (2013). Automatic Detection of Search Tactic in Individual Information Seeking: A Hidden Markov Model Approach. *Proceedings of the iConference 2013*, 712–716.

Hassan, A., Jones, R., and Klinkner, K. L. (2010). Beyond DCG: User Behavior as a Predictor of a Successful Search. In *Proceeding of the 3rd ACM Conference on Web Search and Data Mining*, pages 221–230.

Hauskrecht, M. (2000). Value-Function Approximations for Partially Observable Markov Decision Processes. *Journal of Artificial Intelligence Research*, 13:33–94.

He, Y. and Wang, K. (2011). Inferring Search Behaviors Using Partially Observable Markov Model with Duration (POMD). In *Proceedings of the 4th ACM Conference on Web Search and Data Mining*, pages 415–424.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Howard, R. A. (1960). *Dynamic Programming and Markov Processes*. Technology Press of Massachusetts Institute of Technology.

Hu, M., Liu, S., Wei, F., Wu, Y., Stasko, J., and Ma, K.-L. (2012). Breaking News on Twitter. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2751–2754. ACM.

Huurdeman, H. C. and Kamps, J. (2014). From Multistage Information-Seeking Models to Multistage Search Systems. In *Proceedings of the 5th Information Interaction in Context Symposium*, pages 145–154. ACM.

Ingwersen, P. (1992). *Information Retrieval Interaction*, volume 246. Taylor Graham London.

Jaeger, M. E., Anthony, S., and Rosnow, R. L. (1980). Who Hears What From Whom and with What Effect: A Study of Rumor. *Personality and Social Psychology Bulletin*, 6(3):473–478.

Jelinek, F., Bahl, L., and Mercer, R. (1975). Design of a Linguistic Statistical Decoder for the Recognition of Continuous Speech. *IEEE Transactions on Information Theory*, 21(3):250–256.

Jin, X., Sloan, M., and Wang, J. (2013). Interactive Exploratory Search for Multi Page Search Results. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 655–666. ACM.

Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. (1998). Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence*, 101(1):99–134.

Kochkina, E., Liakata, M., and Augenstein, I. (2017). Turing at SemEval-2017 Task 8: Sequential Approach to Rumour Stance Classification With Branch-LSTM. *Proceedings of the 11th International Workshop on Semantic Evaluation*, pages 475–480.

Kotzyba, M., Gossen, T., Schwerdt, J., and Nürnberger, A. (2017). Exploration or Fact-Finding: Inferring User's Search Activity Just in Time. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, pages 87–96. ACM.

Kuhlthau, C. C. (1991). Inside the Search Process: Information Seeking From the User's Perspective. *Journal of the American Society for Information Science*, 42(5):361–371.

Kwon, S., Cha, M., Jung, K., Chen, W., and Wang, Y. (2013). Prominent Features of Rumor Propagation in Online Social Media. In *Proceedings of the 13th International Conference on Data Mining*, pages 1103–1108. IEEE.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289. Morgan Kaufmann Publishers Inc.

Lane, T. (1999). Hidden Markov Models for Human/Computer Interface Modeling. In *Proceedings of the IJCAI-99 Workshop on Learning About Users*, pages 35–44. Citeseer.

Längkvist, M., Karlsson, L., and Loutfi, A. (2014). A Review of Unsupervised Feature Learning and Deep Learning for Time-Series Modeling. *Pattern Recognition Letters*, 42:11–24.

Liporace, L. (1982). Maximum Likelihood Estimation for Multivariate Observations of Markov Sources. *IEEE Transactions on Information Theory*, 28(5):729–734.

Lipton, Z. C., Berkowitz, J., and Elkan, C. (2015). A Critical Review of Recurrent Neural Networks for Sequence Learning. *arXiv preprint arXiv:1506.00019*.

Liu, X., Nourbakhsh, A., Li, Q., Fang, R., and Shah, S. (2015a). Real-Time Rumor Debunking on Twitter. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1867–1870. ACM.

Liu, Y., Gao, B., Liu, T.-Y., Zhang, Y., Ma, Z., He, S., and Li, H. (2008). BrowseRank: Letting Web Users Vote for Page Importance. In *Proceedings of the 31st SIGIR Conference on Research and Development in Information Retrieval*, pages 451–458. ACM.

Liu, Y.-Y., Li, S., Li, F., Song, L., and Rehg, J. M. (2015b). Efficient Learning of Continuous-Time Hidden Markov Models for Disease Progression. In *Advances in Neural Information Processing Systems*, pages 3600–3608.

Lukasik, M., Cohn, T., and Bontcheva, K. (2015). Classifying Tweet Level Judgements of Rumours in Social Media. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2590–2595.

Lukasik, M., Srijith, P., Vu, D., Bontcheva, K., Zubiaga, A., and Cohn, T. (2016). Hawkes Processes for Continuous Time Sequence Classification: An Application to Rumour Stance Classification in Twitter. In *Proceedings of 54th Annual Meeting of the Association for Computational Linguistics*, pages 393–398. Association for Computational Linguistics.

Luo, J., Zhang, S., Dong, X., and Yang, H. (2015). Designing States, Actions, and Rewards for Using POMDP in Session Search. In *Advances in Information Retrieval: 37th European Conference on IR Research*, pages 526–537. Springer Int. Publishing.

Luo, J., Zhang, S., and Yang, H. (2014). Win-Win Search: Dual-Agent Stochastic Game in Session Search. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 587–596. ACM.

Ma, B., Lin, D., and Cao, D. (2017). Content Representation for Microblog Rumor Detection. In *Advances in Computational Intelligence Systems*, pages 245–251. Springer.

Ma, J., Gao, W., Wei, Z., Lu, Y., and Wong, K.-F. (2015). Detect Rumors Using Time Series of Social Context Information on Microblogging Websites. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1751–1754. ACM.

Marchionini, G. (1997). *Information Seeking in Electronic Environments*. Cambridge University Press.

Markov, A. (1906). Extension of the Law of Large Numbers to Quantities, Depending on Each Other (1906). Reprint. *Journal Électronique d'Histoire des Probabilités et de la Statistique*, 2(1b):Article 10, 12 pages, electronic only article.

Meho, L. I. and Tibbo, H. R. (2003). Modeling the Information-Seeking Behavior of Social Scientists: Ellis's Study Revisited. *Journal of the American Society for Information Science and Technology*, 54(6):570–587.

Meinert, J., Mirbabaie, M., Dungs, S., and Aker, A. (2018). Is it Really Fake?—Towards an Understanding of Fake News in Social Media Communication. In *International Conference on Social Computing and Social Media*, pages 484–497. Springer.

Mendoza, M., Poblete, B., and Castillo, C. (2010). Twitter Under Crisis: Can we trust what we RT? In *Proceedings of the 1st Workshop on Social Media Analytics*, pages 71–79. ACM.

Mohammad, S., Kiritchenko, S., Sobhani, P., Zhu, X., and Cherry, C. (2016). SemEval-2016 Task 6: Detecting Stance in Tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, pages 31–41.

Ng, A. Y. and Jordan, M. I. (2002). On Siscriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. In *Advances in Neural Information Processing Systems*, pages 841–848.

Pharo, N. and Nordlie, R. (2012). Examining the Effect of Task Stage and Topic Knowledge on Searcher Interaction With a 'Digital Bookstore'. In *Proceedings of the 4th Information Interaction in Context Symposium*, pages 4–11. ACM.

Ponomareva, N., Rosso, P., Pla, F., and Molina, A. (2007). Conditional Random Fields vs. Hidden Markov Models in a Biomedical Named Entity Recognition Task. In *Proceedings of International Conference on Recent Advances in Natural Language Processing*, pages 479–483.

Powers, D. M. (2011). Evaluation: From Precision, Recall and F-measure to ROC, Informedness, Markedness and Correlation.

Procter, R., Vis, F., and Voss, A. (2013). Reading the Riots on Twitter: Methodological Innovation for the Analysis of Big Data. *International Journal of Social Research Methodology*, 16(3):197–214.

Puterman, M. L. (1990). Markov Decision Processes. In *Stochastic Models*, volume 2 of *Handbooks in Operations Research and Management Science*, chapter 8, pages 331 – 434. Elsevier.

Puterman, M. L. (2014). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons.

Qazvinian, V., Rosengren, E., Radev, D. R., and Mei, Q. (2011). Rumor Has It: Identifying Misinformation in Microblogs. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1589–1599.

Rabiner, L. R. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257–286.

Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks. *arXiv preprint arXiv:1511.06434*.

Rosenblatt, F. (1957). *The Perceptron, a Perceiving and Recognizing Automaton (Project Para)*. Cornell Aeronautical Laboratory.

Rubin, V. L. (2017). Deception Detection and Rumor Debunking for Social Media. *The SAGE Handbook of Social Media Research Methods*, pages 342–363.

Sadikov, E., Madhavan, J., Wang, L., and Halevy, A. (2010). Clustering Query Refinements by User Intent. In *Proceedings of the 19th International Conference on World Wide Web*, pages 841–850. ACM.

Sarawagi, S. and Cohen, W. W. (2005). Semi-Markov Conditional Random Fields for Information Extraction. In *Advances in Neural Information Processing Systems*, pages 1185–1192.

Settles, B. (2004). Biomedical Named Entity Recognition Using Conditional Random Fields and Rich Feature Sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, pages 104–107. Association for Computational Linguistics.

Singh, V., Narayan, S., Akhtar, M. S., Ekbal, A., and Bhattacharyya, P. (2017). IITP at SemEval-2017 Task 8: A Supervised Approach for Rumour Evaluation.

In *Proceedings of the 11th International Workshop on Semantic Evaluation*, pages 497–501.

Spink, A., Greisdorf, H., and Bateman, J. (1998). From Highly Relevant to Not Relevant: Examining Different Regions of Relevance. *Information Processing & Management*, 34(5):599–621.

Srivastava, A., Rehm, G., and Schneider, J. M. (2017). DFKI-DKT at SemEval-2017 Task 8: Rumour Detection and Classification Using Cascading Heuristics. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, pages 486–490.

Stratonovich, R. L. (1960). Conditional Markov Processes. *Theory of Probability & Its Applications*, 5(2):156–178.

Åström, K. (1965). Optimal Control of Markov Processes with Incomplete State Information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205.

Sutton, C. and McCallum, A. (2006). *An Introduction to Conditional Random Fields for Relational Learning*, volume 2. Introduction to Statistical Relational Learning. MIT Press.

Tokuda, K., Masuko, T., Miyazaki, N., and Kobayashi, T. (1999). Hidden Markov Models Based on Multi-Space Probability Distribution for Pitch Pattern Modeling. In *Proceedings of the 1999 International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 229–232. IEEE.

Tokuda, K., Masuko, T., Miyazaki, N., and Kobayashi, T. (2002). Multi-Space Probability Distribution HMM. *Institute of Electronics, Information and Communication Engineers Transactions on Information and Systems*, 85(3):455–464.

Tokuda, K., Nankaku, Y., Toda, T., Zen, H., Yamagishi, J., and Oura, K. (2013). Speech Synthesis Based on Hidden Markov Models. *Proceedings of the IEEE*, 101(5):1234–1252.

Tran, V. and Fuhr, N. (2018). Personalised Session Difficulty Prediction in an Online Academic Search Engine. In *International Conference on Theory and Practice of Digital Libraries*, pages 174–185. Springer.

Tran, V., Maxwell, D., Fuhr, N., and Azzopardi, L. (2017). Personalised Search Time Prediction Using Markov Chains. In *Proceedings of the 3rd ACM International Conference on the Theory of Information Retrieval*. ACM.

Tran, V. T. and Fuhr, N. (2012). Using Eye-Tracking with Dynamic Areas of Interest for Analyzing Interactive Information Retrieval. In *Proceedings of the*

35th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1165–1166.

Tran, V. T. and Fuhr, N. (2013). Markov Modeling for User Interaction in Retrieval. In *SIGIR 2013 Workshop on Modeling User Behavior for Information Retrieval Evaluation*, pages 1–2.

Twardowski, B. (2016). Modelling Contextual Information in Session-Aware Recommender Systems with Neural Networks. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pages 273–276. ACM.

Vakkari, P. (2001). A Theory of the Task-Based Information Retrieval Process: A Summary and Generalisation of a Longitudinal Study. *Journal of Documentation*, 57(1):44–60.

Venter, J. C., Adams, M. D., Myers, E. W., Li, P. W., Mural, R. J., Sutton, G. G., Smith, H. O., Yandell, M., Evans, C. A., Holt, R. A., et al. (2001). The Sequence of the Human Genome. *Science*, 291(5507):1304–1351.

Viterbi, A. (1967). Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269.

Vosoughi, S. (2015). *Automatic Detection and Verification of Rumors on Twitter*. PhD thesis.

Wang, F., Lan, M., and Wu, Y. (2017). ECNU at SemEval-2017 Task 8: Rumour Evaluation Using Effective Features and Supervised Ensemble Models. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, pages 491–496.

Wang, K., Gloy, N., and Li, X. (2010). Inferring Search Behaviors Using Partially Observable Markov (POM) Model. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, pages 211–220. ACM.

Wang, Y., Loe, K.-F., and Wu, J.-K. (2006). A Dynamic Conditional Random Field Model for Foreground and Shadow Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(2):279–289.

Wilson, T. (1999). Models in Information Behaviour Research. *Journal of Documentation*, 55(3):249–270.

Witten, I. H., Frank, E., Hall, M. A., and Pal, C. J. (2016). *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann.

Wojek, C. and Schiele, B. (2008). A Dynamic Conditional Random Field Model for Joint Labeling of Object and Scene Classes. In *European Conference on Computer Vision*, pages 733–747. Springer.

Wu, K., Yang, S., and Zhu, K. Q. (2015). False Rumors Detection on Sina Weibo by Propagation Structures. In *Proceedings of the 31st IEEE International Conference on Data Engineering*, pages 651–662. IEEE.

Xie, I. and Joo, S. (2010). Transitions in Search Tactics During the Web-based Search Process. *Journal of the American Society for Information Science and Technology*, 61(11):2188–2205.

Yang, F., Liu, Y., Yu, X., and Yang, M. (2012). Automatic Detection of Rumor on Sina Weibo. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, pages 1–7. ACM.

Yang, G. H., Dong, X., Luo, J., and Zhang, S. (2018). Session Search Modeling by Partially Observable Markov Decision Process. *Information Retrieval Journal*, 21(1):56–80.

Yang, Q. and Wu, X. (2006). 10 Challenging Problems in Data Mining Research. *International Journal of Information Technology & Decision Making*, 5(04):597–604.

Yue, Z., Han, S., and He, D. (2014). Modeling Search Processes Using Hidden States in Collaborative Exploratory Web Search. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pages 820–830. ACM.

Zhang, S., Luo, J., and Yang, H. (2014). A POMDP Model for Content-Free Document Re-Ranking. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1139–1142. ACM.

Zhao, Z., Resnick, P., and Mei, Q. (2015). Enquiring Minds: Early Detection of Rumors in Social Media from Enquiry Posts. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1395–1405.

Zubiaga, A., Aker, A., Bontcheva, K., Liakata, M., and Procter, R. (2018). Detection and Resolution of Rumours in Social Media: A Survey. *ACM Computing Surveys (CSUR)*, 51(2):1–36.

Zubiaga, A., Liakata, M., and Procter, R. (2017). Exploiting Context for Rumour Detection in Social Media. In *International Conference on Social Informatics*, pages 109–123. Springer.

Zubiaga, A., Liakata, M., Procter, R., Wong Sak Hoi, G., and Tolmie, P. (2016). Analysing How People Orient to and Spread Rumours in Social Media by Looking at Conversational Threads. *PLoS ONE*, 11(3):1–29.