Technische Universität Berlin

Quality and Usability Lab

Part-of-Speech Tagging with Neural Networks for a Conversational Agent

Master Thesis

Master of Science (M.Sc.)

Author	Andreas Müller
Major	Computer Engineering
Matriculation No.	333471

Date	18th May 2018
1st supervisor	Prof. Dr.-Ing. Sebastian Möller
2nd supervisor	Dr. Axel Küpper

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Alle Ausführungen, die anderen veröffentlichten oder nicht veröffentlichten Schriften wörtlich oder sinngemäß entnommen wurden, habe ich kenntlich gemacht.

Die Arbeit hat in gleicher oder ähnlicher Fassung noch keiner anderen Prüfungsbehörde vorgelegen.

Berlin, den April 22, 2018

_____

Unterschrift

Abstract

A part-of-speech tagger is a system which automatically assigns the part of speech to words using contextual information. Potential applications for part-of-speech taggers exist in many areas of computational linguistics including speech recognition, speech synthesis, machine translation or information retrieval in general.

The part-of-speech tagging task of natural language processing is also used in the advisory artificial conversational agent called ALEX. ALEX was developed to answer questions about modules and courses at the Technische Universität Berlin. The system takes the written natural language requests from the user and tries to transform them into SQL-queries. To understand the natural language queries, the system uses a Hidden Markov Model (HMM) to assign tags to each word of the query (part-of-speech tagging). This HMM tagger is trained with manually created training templates that are filled with the data in the database to be queried. The manually created sentence-templates and the slot-filling resulted in many training data sentences with the same structure. This often led to wrong tagging results when the HMM tagger was presented with an input sentence, having a structure that doesn't occur in the training templates.

This thesis shows two different neural network approaches for the language modeling of the input sentences and evaluates and compares both neural network based tagger as well as the HMM based tagger.

Zusammenfassung

Ein Part-of-speech Tagger ist ein System, welches Wortarten anhand von Kontextinformationen automatisch den gegebenen Wörtern zuordnet. Potentielle Anwendungen solcher Tagger gibt es in vielen Bereichen der Computerlinguistik wie Spracherkennung, Sprachsynthese, maschinelle Übersetzung oder Information Retrieval im Allgemeinen.

Part-of-speech Tagging wird auch in ALEX verwendet, einem Artificial Conversational Agent. ALEX wurde entwickelt, um Fragen zu Modulen und Lehrveranstaltungen an der Technischen Universität Berlin zu beantworten. Das System nimmt die in natürlicher Sprache geschriebenen Anfragen des Benutzers und versucht diese in SQL-Abfragen umzuwandeln. Um die natürliche Sprache zu verstehen, verwendet das System ein Hidden-Markov-Model (HMM), um jedem Wort der Eingabe Wortarten zuzuweisen (Part-of-speech Tagging). Dieser HMM-Tagger wird mit manuell erstellten Trainingsvorlagen trainiert, die mit den Daten der abzufragenden Datenbank gefüllt werden. Die manuell erstellten Satzvorlagen führten zu vielen Trainingsdatensätzen mit gleicher Struktur und damit oft zu falschen Tagging-Ergebnissen, wenn der HMM-Tagger einen Eingabesatz mit einer Struktur verarbeiten sollte, die in den Trainingsvorlagen nicht vorkommt.

Diese Arbeit zeigt zwei verschiedene Ansätze für die Sprachmodellierung der Eingabesätze basierend auf neuronalen Netzwerken und bewertet und vergleicht sowohl die Neuronalen Netzwerk-basierten Tagger als auch den HMM-basierten Tagger.

Contents

List of Figures

List of Tables

Abbreviations

ACA	*Artificial Conversational Agent*
FNN	*Feed-forward Neural Network*
HMM	*Hidden Markov Model*
NLP	*Natural Language Processing*
NLTK	*Natural Language Toolkit*
RNN	*Recurrent Neural Network*

1 Introduction

1.1 Scope of this Thesis

The scope of this thesis is the development of a neural network based part-of-speech tagger for the advisory Artificial Conversational Agent (ACA) ALEX, the training of different language models and their evaluation with corresponding test sets.

In order to accomplish the new language models, two different neural network architectures are implemented: A feed-forward neural network and a recurrent neural network. For the training of both neural network implementations, a corpus of tagged language data is generated with the help various input templates, which are created on the basis of logged user input data.

To evaluate the language models, a data set of known data[1] and unknown data[2] is created. On the basis of this evaluation, both neural network models and the HMM are compared to each other.

In accordance to the evaluation results, the former HMM based part-of-speech tagger is then replaced by this new tagger. To guarantee a seamless integration, the new tagger is implemented as a separate module with the same program interface the old tagger already utilizes. This way no other components of the conversational agent have to be changed and the effort of the replacement is kept minimal.

1.2 Related Work

This thesis is build upon the work of T. Michael [8], who describes the design and implementation of ALEX in detail. The conversational agent was implemented for the purpose of helping students of the TU Berlin to organize their studies by providing a

1 Data, that was already used for the training of the model
2 Data, that includes words and sentence structures, that didn't occur in the training data sets

simple way to gain information about modules and courses. It utilizes two separate already existing baseline systems by merging their data into one relational database. This database is used as the central access point for the information that users want to retrieve.

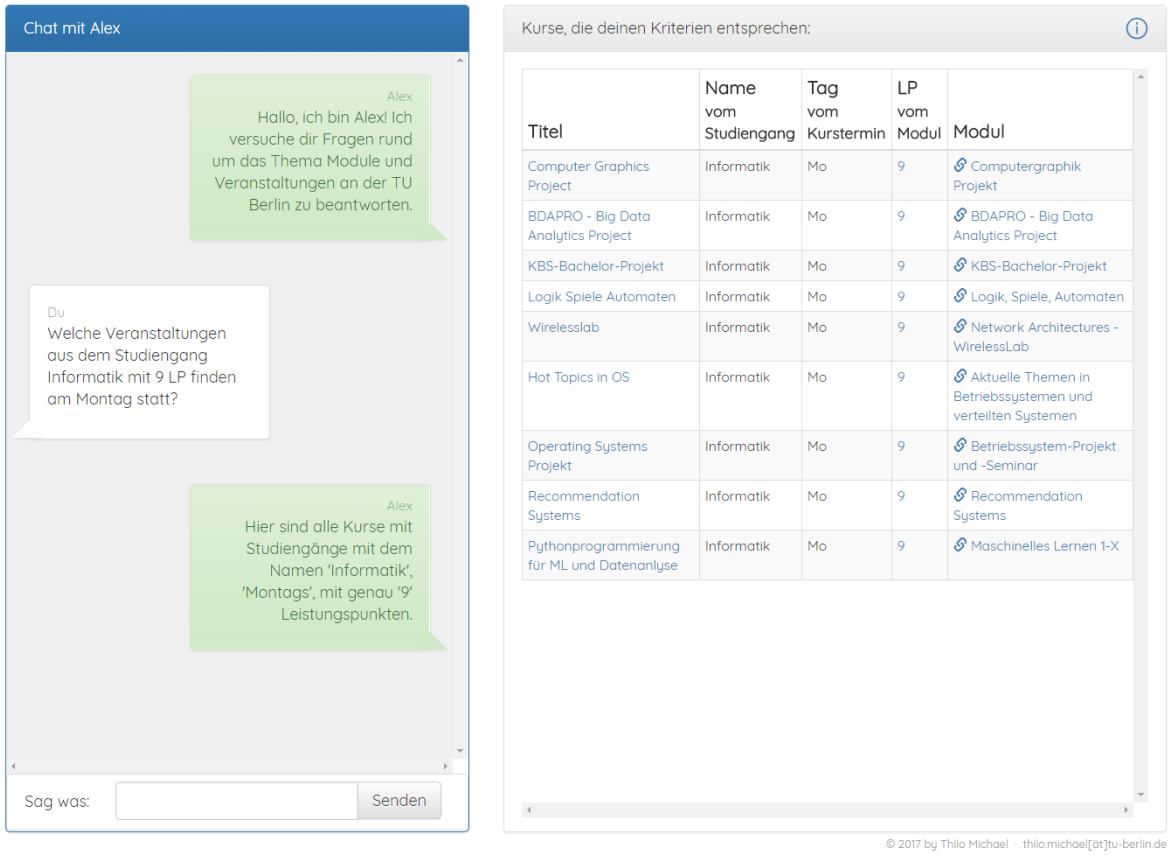

Figure 1.1: The user interface of ALEX. The left section contains the conversation with the Agent and a field where the user can type. The right section shows the result of the generated database query in tabular form.
The user asked for all courses of the subject *computer science* that provide 9 ECTS and are scheduled on a Monday. The agent answered accordingly.
This image was captured at the 21st April 2018.

ALEX consists of several processing modules:

- The **tagging module** uses a Hidden Markov Model to calculate the parts of speech for the user input, later described in chapter 2.2
- The **query generation module** composes actual SQL queries from the tagging output data by recognizing the requested model and the return type

- The **filter extraction module** provides refinement and constraint handling for the query generator
- The **response generation module** formulates answers for the user input in natural language by processing the generated query, the recognized model and the conversation state.

Moreover ALEX provides a user interface which utilizes web technologies and can be accessed in a web browser. Figure 1.1 shows the user interface where the user asked one question and the agent returned the result in tabular form and answered accordingly.

The focus of this thesis lies on the tagging module, as the main objective is to replace the Hidden Markov Model by Artificial Neural Networks.

1.2.1 The Hidden Markov Model

The Hidden Markov Model (HMM) is a probabilistic finite state machine that solves classification problems in general. It uses the observable output data of a system to derive hidden information from it. Among other applications, HMMs are used especially for speech recognition tasks.

The preliminary work for HMMs was done by R. L. Stratonovich. He first described the conditional Markov processes in 1960 [14] that were used in the following years to describe simple Markov Models and later Hidden Markov Models (see Baum et. al. [3][2]). The latter became popular for solving the task of automatic recognition of continuous speech [1] along with other applications like pattern recognition in general, the analysis of biological sequences (e.g. DNA) [4] and part-of-speech tagging.

1.2.2 The Artificial Neural Network Model

Artificial Neural Networks are networks that process information inspired by biological nervous system. They consist of connected computational units typically arranged in different layers. Such a unit (also called *artificial neuron*) can make

calculations based on its inputs and pass the result to the next connected units. These connections are weighted, so that the weight can be adjusted depending on the activity of the unit. Thus a model of the features of the input data can be created.

After preceding research by W. McCulloch, W. Pitts [7] and D. Hebb [13] about arithmetical learning methods inspired by the connections of neurons in the 1940s, M. Minsky built the first neural network learning machine called SNARC (*Stochastic Neural Analog Reinforcement Computer)*[5] in 1951.

In the late 1950s, F. Rosenblatt developed the *Mark I Perceptron* computer and published a theorem of convergence of the perceptron[11] in 1958. He coined the term *perceptron* for an algorithm that was able to learn the assignment of input data to different classes. The perceptron represents a simple artificial neural network containing one single neuron at first[3]. F. Rosenblatt stated, that every function that is representable by the model can be learned with the proposed learning method. In 1960, B. Widrow presented the ADALINE[4] model of a neural network, where the input weights could already be adjusted by the learning algorithm [15].

A publication of M. Minsky and S. Papert [9] in 1969 analyzed and exposed some significant limitations of the basic perceptron. They pointed out, that it is not possible to learn functions without linear separability (e.g. the exclusive-or problem). Due to these limitations and the fact, that the processing power of computers at that time was not sufficient for larger neural networks, the research interest in artificial neural networks decreased in the following years.

In 1982, J. Hopfield presented a previously described Neural Network with feedback (known as *Hopfield network*), that was able to solve optimization problems like the *Travelling Salesman Problem*[5].

3 Chapters 3.1 and 3.2 explain the architecture of different neural network structures in detail

4 ADALINE is an acronym for Adaptive Linear Neuron

5 The problem of the travelling salesman or round trip problem: The order of places to be visited once should be chosen in such a way that the distance covered is minimal, whereby the last place is again the starting point (round trip).

1.3 Structure of this Thesis

As introduction, this first chapter gave a short overview about the subject of natural language processing and part-of-speech tagging in general.

The second chapter describes structure and functionality of the already existing ACA ALEX with the main focus on its language model and tagging interface.

Chapter 3 explains the implementation of a part-of-speech tagging system with two different neural network approaches.

The training of the language models including the retrieval of the training data and tuning of the training parameter is described in Chapter 4.

Chapter 5 shows the evaluation of each language model with a generated test set and their comparison.

In conclusion the final Chapter 6 discusses and summarizes the evaluation results and gives an outlook on future work.

2 ALEX: Artificial Conversational Agent

...

2.1 System Overview

The modular structure of ALEX allows the separation of different functions and therefore easier replaceability of certain functionalities. Besides a web crawler for current data retrieval for the database and a frontend interface module, ALEX offers a tagging module. This module provides the training of a language model as well as the assignment of tags to the words of a given input sentence.

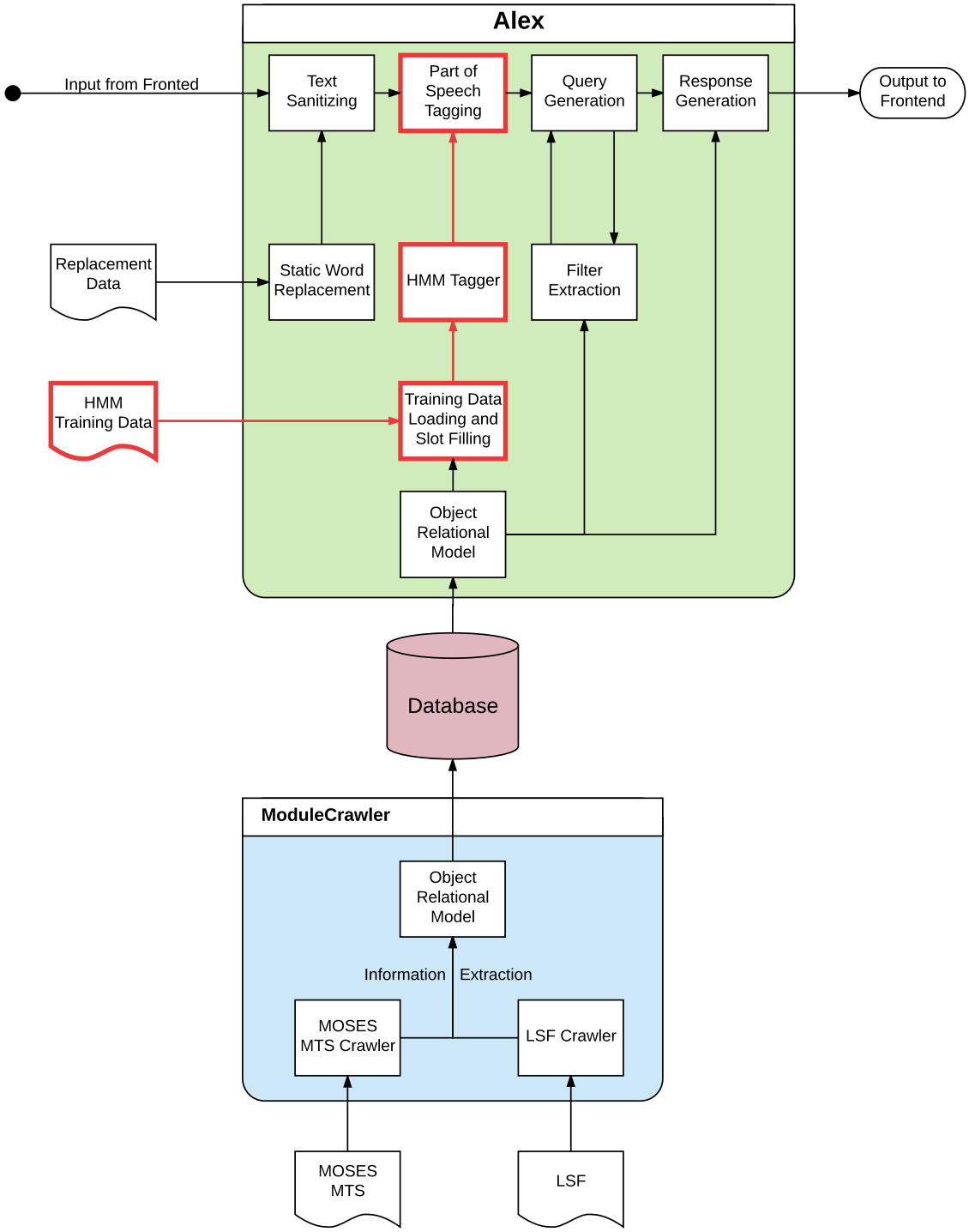

Figure 2.1: Overview of all components of ALEX. The red colored parts are components that lie within the scope of this thesis and are adapted or replaced. Original figure by T. Michael [8]

2.2 The Hidden Markov Model Tagger

As described in section 1.2.1 of the introduction, the Hidden Markov Model (HMM) is a statistical tool that uses observable output data of a system to derive hidden information from it. Applications are image processing, gesture recognition and natural language processing tasks like speech recognition and part-of-speech tagging in particular.

In case of POS tagging, the observable states of the HMM represent the given sequence of words whereas the hidden states represent the corresponding parts of speech. The HMM calculates the joint probability of the whole sequence of hidden states with the help of transmission and output probabilities. Subsequently it finds the maximum probability of all possible state sequences and decides as a result, which parts of speech are most likely applied to the words of the input sequence.

Figure 2.2 illustrates an example of a state sequence with three hidden states (part of speech tags) and the observed word sequence in an HMM. The calculation of the joint probability P of the word sequence in this case is shown in equation 2.1 as the product of transmission and output probabilities.

$$P = p_{start} \cdot p_{out,1} \cdot p_{trans,1} \cdot p_{out,2} \cdot p_{trans,2} \cdot p_{out,3} \tag{2.1}$$

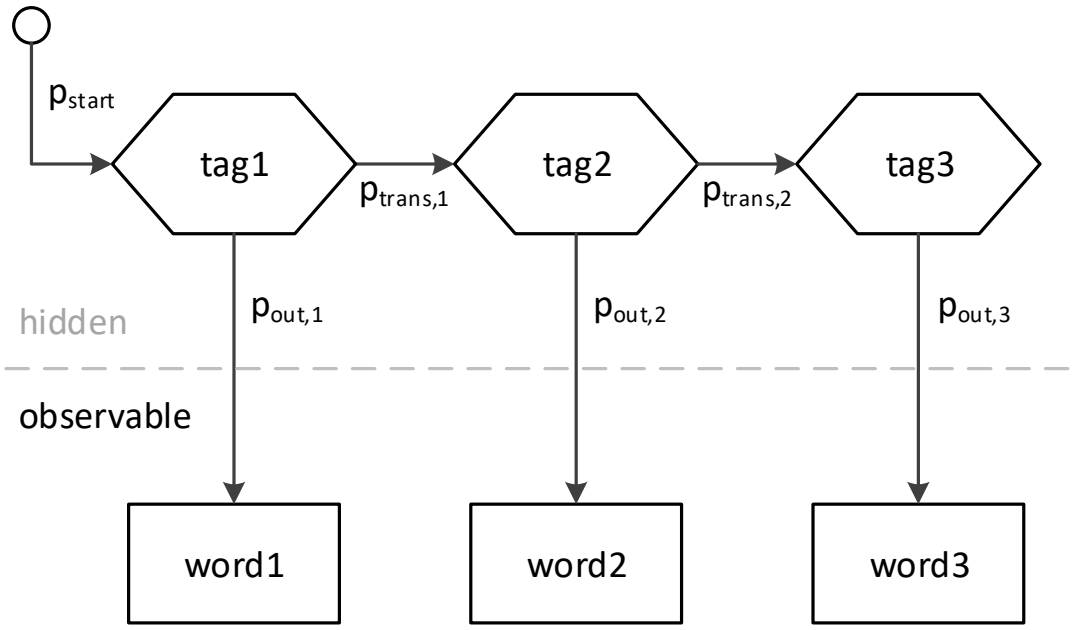

Figure 2.2: An example of a state sequence of three hidden states (tag1 – tag3) and an observed sequence of three words (word1 – word3) in an Hidden Markov Model. p_{start} denotes the start probability, p_{trans} the transmission probabilities between hidden states and p_{out} the output probabilities between a hidden state and an output.

For this purpose, an HMM is included in ALEX. According to T. Michael [8], a tagging scheme was developed to extract exactly the information from a user input that is needed to successfully create a database query and return the information the user asked for. This tagging scheme was intentionally built domain independent, giving the opportunity to make ALEX an ACA for any topic providing a corresponding database and training data.

To maintain this universal applicability as well as the compatibility to other modules of ALEX, the Neural Network approaches presented in this thesis use the same tagging scheme ALEX already utilizes. For a better understanding of the evaluation results in Chapter 5, table 2.1 gives an overview of the 6 different classes of tags that are used by ALEX.

Class	Formats	Description	Example
R	*Return-tags*, describing data that is expected to be returned	R_LIST R_SINGLE R_COUNT	*"**Which** modules..."* *"**Which** module..."* *"**How many** modules..."*
M	*Model-tags*, describing the database model, e.g. M_MTSModule or M_Course	M_[MODEL]	*"Which **modules**..."* *"Which **courses**..."*
C	*Constraint-tags*, filtering the result set, given a database model and corresponding field, e.g. C_MTSModule:ects	C_[MODEL]:[FIELD]	*"Modules with **6** ects..."*
P	*Property-tags*, indicating to include fields in the result set, e.g. P_MTSModule:ects	P_[MODEL]:[FIELD]	*"Modules with 6 **ects**..."*
Q	*Comparison-tags*, describing an equal, greater than or less than constraint	Q_EQ Q_LT Q_GT	*"...with **exactly** 6 ects..."* *"...**less than** 6 ects..."* *"...**more than** 6 ects..."*
X	*Extra-tags*, describing words that are not relevant for the database query[1]	X X_[WORD] X_[MODEL]:[FIELD]	*"**and**", "**of**", "**is**"* *"I need **help**"* *"**Professor** John Doe"*

Table 2.1: Overview of the tagging scheme used in ALEX, consisting of 6 different classes of tags with a total of 12 different formats. The examples contain **emphasized** words that belong to the corresponding tag format. Detailed explanation of the tagging classes and its formats is given by T. Michael [8].

2.3 Tagging Interface

As described in the previous chapter, the implementation of the tagging module of ALEX utilizes a Hidden Markov Model for the part-of-speech tagging. ALEX uses

1 This can be either words with no special meaning at all (tagged with X), or words that have no meaning for the database query but for the system itself (e.g. the tag X_HELP for the word *"help"*) or words that lead to a particular constraint (like the tag X_Person:fullname for the word *"Professor"*, that leads to a name).

an already existing implementation of the HMM Tagger from the Natural Language Toolkit (NLTK)[2], called `HiddenMarkovModelTagger`.

To replace the existing tagger, a new tagger has to provide a class with two methods: `train` and `tag`. These methods are used to create the language model and apply it to unknown data.

The `train` method creates a new instance of the tagger class, trains this class with the given training data and returns it. The training data itself must be a list of sentences, where a sentence is a list of tuples, containing each word of this sentence and its corresponding tag. The following exemplifies the structure of the training input data containing two sentences where each word is tagged with *TAG*:

```
[
  [ ('the', TAG), ('dog', TAG), ('is', TAG), ('running', TAG) ],
  [ ('the', TAG), ('cat', TAG), ('sleeps', TAG), ('all', TAG), ('day', TAG) ]
]
```

The `tag` method attaches a tag to each word of an input sentence, according to the previously trained language model. The input has to be an unknown sentence as a simple list of words:

```
[ 'an', 'unknown', 'test', 'sentence' ]
```

The output is a corresponding list of tuples containing a word and its assigned tag:

```
[ ('an', TAG), ('unknown', TAG), ('test', TAG), ('sentence', TAG) ]
```

2 The Natural Language Toolkit is a collection of *Python* programming libraries for natural language processing, see `http://nltk.org`

3 Part-of-Speech Tagging

... Ma et. al. have shown before, that the neural network approach outperforms part-of-speech tagger that are based on statistical models (like the HMM) [6].

3.1 Feed-forward Neural Network Model

...

3.1.1 Architecture

...

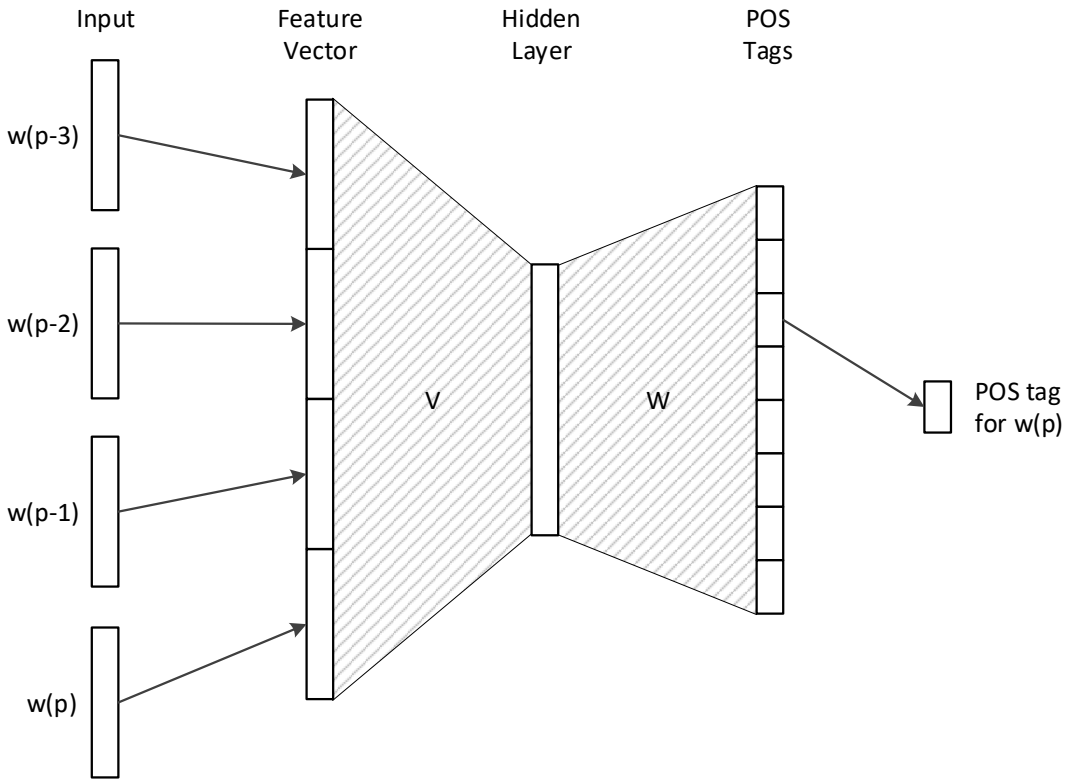

Figure 3.1: The structure of a feed-forward neural network. The feature vector is built by the initial vectors of the corresponding input word on position p and by its three predecessors (here as an example).

3.1.2 Implementation

...

3.2 Recurrent Neural Network Model

...

3.2.1 Architecture

...

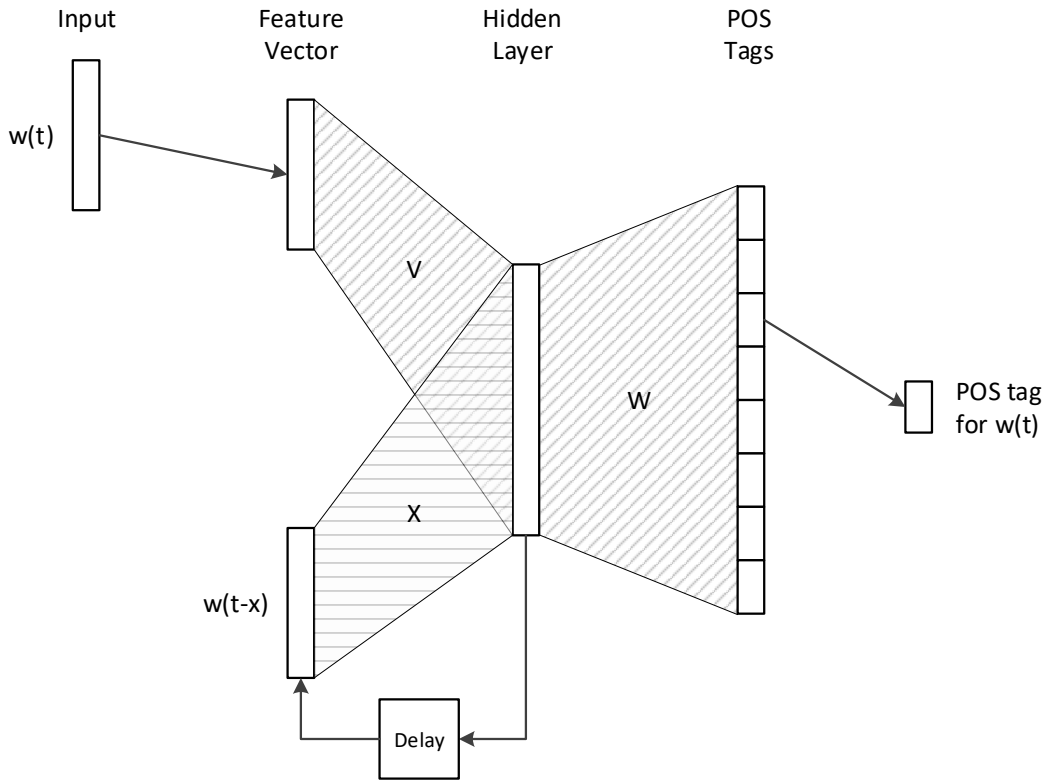

Figure 3.2: The structure of a recurrent neural network. The feature vector is the initial vector of the corresponding input word at time `t`. The output of the hidden layer from previously trained words (here as an example at time `x`) is fed back into the same hidden layer for the current word.

3.2.2 Implementation

...

4 Training

...

4.1 Data Retrieval

...

4.2 Parameter Tuning

...

5 Evaluation and Comparison

...

5.1 Test Design

...

6 Discussion and Conclusion

...

6.1 Summary

...

6.2 Discussion

...

6.3 Future work

... [8] [10] [6] [12]

Bibliography

[1] J. Baker. The DRAGON system–An overview. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):24–29, February 1975.

[2] Leonard E. Baum. An Equality with Applications to Statistical Prediction for Functions of Markov Process and to a Model to Ecology. *Bulletin of the American Mathematical Society*, 73:360–363, 1967.

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

[4] M.J. Bishop and E.A. Thompson. Maximum likelihood alignment of DNA sequences. *Journal of Molecular Biology*, 190(2):159 – 165, 1986.

[5] Daniel Crevier. AI: The Tumultuous Search for Artificial Intelligence. 1993.

[6] Qing Ma, Kiyotaka Uchimoto, Masaki Murata, and Hitoshi Isahara. Elastic Neural Networks for Part of Speech Tagging. 5, January 2000.

[7] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, December 1943.

[8] Thilo Michael. Design and Implementation of an Advisory Artificial Conversational Agent, October 2016.

[9] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. The MIT Press, 1969.

[10] Andreas Müller. Analyse von Wort-Vektoren deutscher Textkorpora, June 2015.

[11] F. Rosenblatt. The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain. *Psychological Review*, pages 65–386, 1958.

[12] Helmut Schmid. Part-of-Speech Tagging with Neural Networks. October 1994.

[13] G. L. Shaw. Donald Hebb: The Organization of Behavior. In Günther Palm and Ad Aertsen, editors, *Brain Theory*, pages 231–233, Berlin, Heidelberg, 1986. Springer Berlin Heidelberg.

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

[15] B. Widrow. An adaptive "ADALINE" neuron using chemical "Memistors". *Stanford Electronics Laboratories Technical Report*, 1553–2(2), 1960.

A First appendix

A.1 test

...