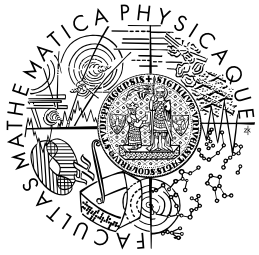

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Abhishek Agrawal

Eye-tracking features in syntactic parsing

Institute of Formal and Applied Linguistics

Supervisor of the master thesis: Mgr. Rudolf Rosa, Ph.D.
Study programme: Computer Science
Study branch: Computational Linguistics

Prague 2020

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

I am very grateful to my friends and family who kept me motivated throughout my time working on this thesis. I would like to thank Mostafa with whom my discussions led me towards my thesis topic. I would also like to thank Altynbek and Memduh for keeping me sane and having all our conversations in times of the quarantine. A special thanks to Maria for kindly providing me with the extracted gaze features without which this work would not have been possible. I am also very grateful to my supervisor Rudolf Rosa who always provided me with valuable advice and insights and for his support throughout my time working on my thesis.

Finally, I am also very grateful to Charles University for granting me the scholarship which funds my study here in Prague.

Title: Eye-tracking features in syntactic parsing

Author: Abhishek Agrawal

Department: Institute of Formal and Applied Linguistics

Supervisor: Mgr. Rudolf Rosa, Ph.D., Institute of Formal and Applied Linguistics

Abstract: In this thesis, we explore the potential benefits of leveraging eye-tracking information for dependency parsing on the English part of the Dundee corpus. To achieve this, we cast dependency parsing as a sequence labelling task and then augment the neural model for sequence labelling with eye-tracking features. We also augment a graph-based parser with eye-tracking features and parse the Dundee Corpus to corroborate our findings from the sequence labelling parser. We then experiment with a variety of parser setups ranging from parsing with all features to a delexicalized parser. Our experiments show that for a parser with all features, although the improvements are positive for the LAS score they are not significant whereas our delexicalized parser significantly outperforms the baseline we established. We also analyze the contribution of various eye-tracking features towards the different parser setups and find that eye-tracking features contain information which is complementary in nature, thus implying that augmenting the parser with various gaze features grouped together provides better performance than any individual gaze feature.

Keywords: eye-tracking features, dependency parsing, delexicalized parsing, sequence labelling

Contents

Introduction

When it comes to natural language processing (NLP), most of the research work that is done is based on textual data that is usually annotated in some way while human language processing data like eye movement recordings, etc. are often overlooked. The solutions for most tasks in NLP these days are based on neural networks which require annotated data on a large scale to learn good representations and provide meaningful results. Annotating such large quantities of data is not an easy task usually requiring a lot of manual labour which is time consuming and can be expensive not to mention resolving the inter-annotator agreement. However, every day, millions of people read their daily newspapers, books, magazines, articles on the internet, etc. in a plethora of languages. In the past couple of years some studies have shown that using behavioral data for NLP tasks involving syntax and semantics can be useful [Mishra et al., 2016a,b,c, Barrett and Søgaard, 2015a,b]. This leads us to believe that if there was a way to tap into cognitive data generated by unconscious human parsing of text, it could be leveraged in some manner to support NLP tools thereby reducing our dependency on annotated textual data.

To this end, we can try to identify certain aspects of human cognitive processing by getting data from several sources like keystroke logging while typing, eye-tracking features while reading, electroencephalography (EEG) signals or functional magnetic resonance imaging (fMRI) scans while reading, etc. which could be potentially used to improve NLP tasks. Eye movement recordings of a person taken when they are reading a text have been known to reflect a person's comprehension of the text [Gaskell et al., 2012]. Various psycholinguistic studies have extensively explored and identified several layers of linguistic processing like syntactic and morphological processing reflected in the eye-tracking data [Frazier and Rayner, 1982, Hyönä et al., 2004]. There are quite a few studies which have found a relation between eye-tracking recordings and the lexical properties of a word. These studies are further described in a later section.

Dependency parsing is a fairly well studied phenomenon because of its relevance in several downstream NLP applications and because of the overwhelming success of the Universal Dependencies (UD) project [Nivre et al., 2016] which is an endeavor to provide guidelines for consistent dependency annotations across multiple languages. This thesis presents an investigation in leveraging eye-tracking features for dependency parsing and the knowledge they bring in identifying syntactic categories and relations. There have been only two previous studies, to the best of our knowledge, which try to leverage gaze features[1] for dependency parsing [Strzyz et al., 2019b, Barrett and Søgaard, 2015b]. Strzyz et al. [2019b] cast dependency parsing as sequence labelling in a multi-task learning setup where the gaze features are predicted as an auxiliary task while considering that the eye-tracking data will be available only during training time. In this thesis, we try an alternative approach to Strzyz et al. [2019b] wherein we focus on determining the effects of incorporating eye-tracking data directly in the model for dependency parsing and assume that eye-tracking data would also be available to

[1]Gaze features are the same as eye-tracking features and the two terms are used interchangeably throughout this thesis

us at inference time as opposed to the approach followed by Strzyz et al. [2019b]. To provide further evidence, we also employ a graph-based parser and augment it with gaze features to see if they improve parsing.

Another reason for choosing to work with eye-tracking features is that it is highly likely that eye-tracking technology will be available on a much larger scale in the near future and hence can be leveraged easily for NLP tasks. This is evidenced by the availability of eye-tracking through regular webcams [San Agustin et al., 2009] and smartphones [Krafka et al., 2016]. Although eye-tracking recording might seem an even more expensive solution to manually annotating data, several studies and even efforts made by the industry as further described in Section 1.2.4 indicate that costs associated with eye-tracking hardware and software are just going to go down and that next generation mass consumer goods like smartphones, tablets, laptops and gaming consoles will come equipped with eye-tracking capabilities.

Research goals

The primary objective of this thesis is not to chase some state-of-the-art score in dependency parsing but to explore the benefits of using eye-tracking features for dependency parsing. We build upon the previous research done in the area of working with gaze features for NLP tasks and incorporate some of the ideas that could potentially improve dependency parsing.

The main questions that the thesis attempts to address can be summarized and enumerated as follows:

1. Do eye-tracking features help in dependency parsing and to what extent?
2. Does delexicalized parsing benefit in any way from gaze features?
3. Can gaze data improve parsing with just lexical features?
4. Which gaze feature contributes the most towards dependency parsing?

Outline

This thesis is divided into several chapters, each of which aims to contribute towards answering the research questions posed by us.

I Background and Related Work

In this chapter, we describe all the relevant topics pertaining to our thesis. We also provide a background on eye-tracking studies, introduce the relevant terminology and provide a review of the prior work on leveraging gaze features for various NLP tasks.

II A Primer on Neural Networks

In this chapter, we provide a brief overview of the different concepts that we use for building our parser starting with a quick primer on neural networks focusing

on those topics which are relevant to our task. We then briefly describe sequence labelling and multi-task learning.

III Parser Descriptions

In this chapter, we describe the two different parsers that we use along with their architectures and hyperparameters.

IV Using Gaze Features for Dependency Parsing

This chapter describes all the experimental setups and their results.

V Evaluation

In this chapter, we evaluate the best setups from the previous chapter on the held out testing data and analyze the results.

Conclusion

In this chapter, we summarize our findings and discuss possible future avenues for further research.

1. Background and Related Work

1.1 Parsing

Parsing a sentence basically means identifying the syntactic structure that is associated with a sentence by trying to figure out the syntactical roles and relations of each word in the sentence. Usually, the syntactic structure is depicted as a tree like structure and hence the name parse trees is given to a parsed sentence.

1.1.1 Dependency Parsing and Constituency Parsing

Constituency parsing

In constituency parsing, a sentence is parsed on the basis of a context-free grammar which was described by Chomsky [1956]. The sentence is recursively broken down into its constituents or sub-phrases until only a single word constituent remains. The Figure 1.1 shows an example of what a constituency parse of a sentence looks like. Note how the terminals of the parse tree are the actual words of the sentence whereas the non-terminals are the constituents or sub-phrases. Some of the algorithms that were widely used for constituency parsing were the Cocke-Kasami-Younger (CKY) algorithm [Kasami, 1965, Younger, 1967], the chart parser [Kaplan, 1973, Kay, 1986] and Earley's algorithm [Earley, 1970]. However, the state-of-the-art results in constituency parsing these days all employ neural networks to some extent [Suzuki et al., 2018, Kitaev and Klein, 2018, Zhou and Zhao, 2019].

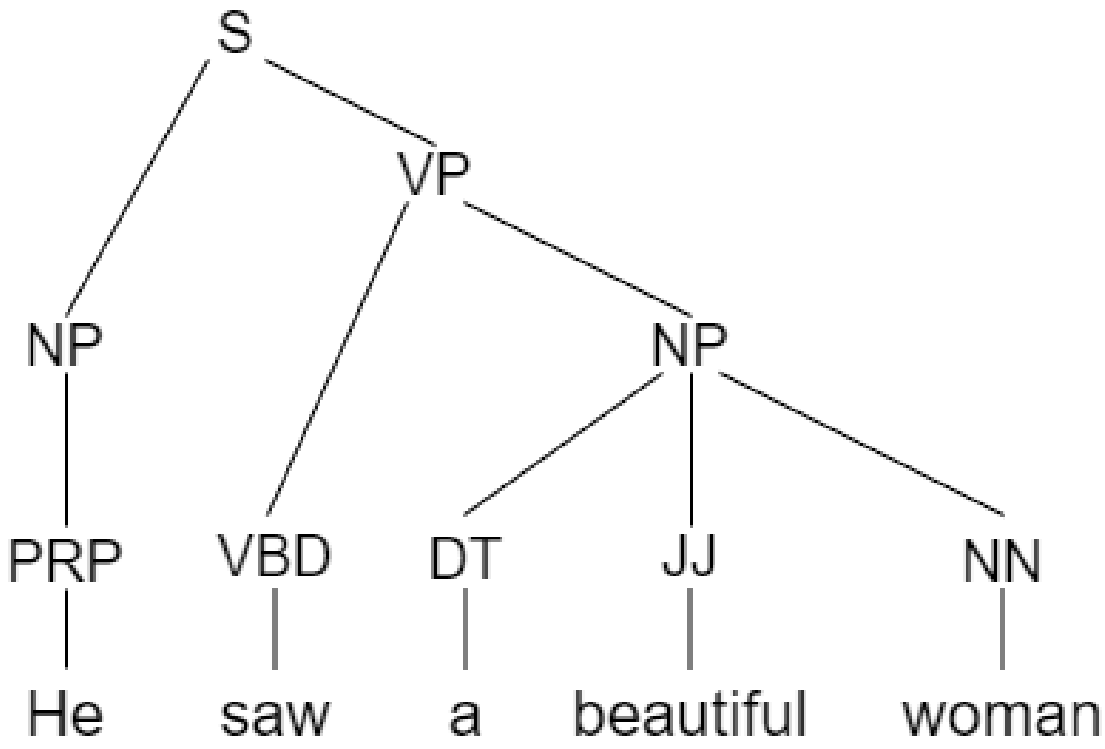

Figure 1.1: A constituency-based parse for a sentence

Dependency parsing

In dependency parsing, a sentence is parsed on the basis of dependency grammar. A dependency parse tree is represented only in terms of the actual words of the sentence and a set of directed grammatical relations between these words. The relations between the words are expressed in the form of directed labelled arcs and these relations are similar to the grammatical relations one learns in his childhood like subject of the verb, object of the verb, etc. The direction of the arc between two words indicates which word is the head and which word is the dependent. One must also note that each word can have only one head although a word can be head to several other words. There is also usually an explicit root node in the parse tree. Dependency trees are more easier to read and understand the relations between words and thus are quite useful for some NLP tasks. Although these relations can also be obtained from constituency trees, they need to be further processed to obtain these relations. In case of free word order languages, it also easier to represent them as dependency trees because in case of constituency trees, a separate rule will be required for each possible position of the word whereas in dependency parsing you will simply have an arc between the words indicating their relation. Covington [2001] provided the foundation for modern transition-based parsers by going through the words one at a time and Nivre [2003] built upon this to provide the arc standard parser. McDonald et al. [2005] introduced graph-based parsers and today's state-of-the-art dependency parsers employ neural networks with usually either a transition-based or graph-based parser [Ji et al., 2019, Dozat and Manning, 2016, Fernández-González and Gómez-Rodríguez, 2019]. In this thesis, we work with only dependency parsing. Figure 1.2 shows the dependency parse for the same sentence from Figure 1.1.

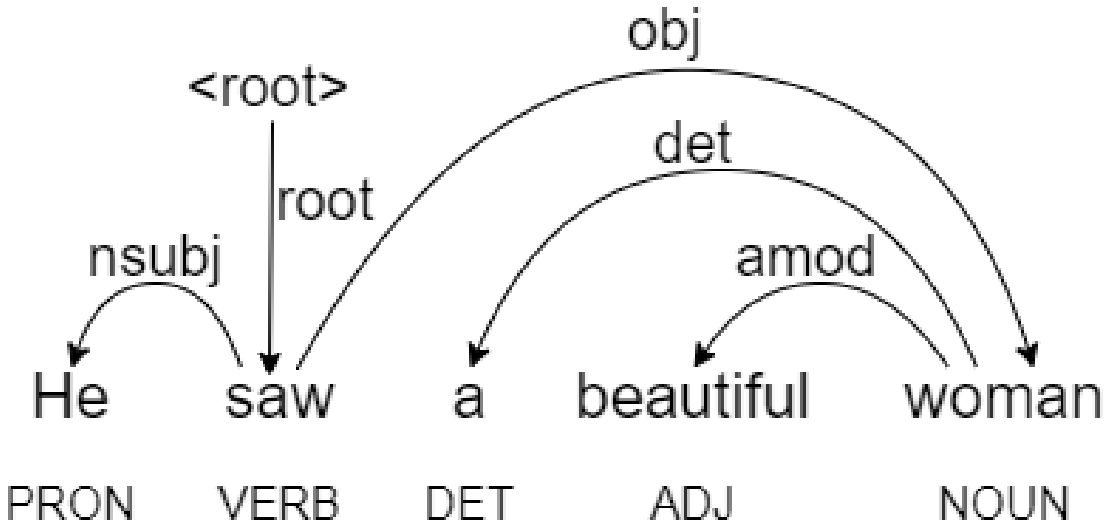

Figure 1.2: A dependency parse for a sentence

1.1.2 Types of Dependency Parsers

Transition-based parser

The idea of transition-based parsing was born by looking at how programming languages or formal languages were parsed. Similar to parsing a programming language, a transition-based parser makes use of three things, namely, a stack of partially constructed trees, an input buffer which contains the words/tokens of a sentence and a set of dependency relations. In most of the transition-based parsers a classifier is trained to play the role of an oracle which is an algorithm that has access to the gold parse trees and tells the parser the correct action to take at each step. However, in practice one doesn't have access to the gold parse trees when actually parsing the sentences and so the classifier is trained by feeding it the data from existing treebanks which helps it learn what actions to take give the state of the stack and the input buffer.

Initially the stack contains only the root node and the input buffer contains all the words of the sentence. Then looking at the stack and the input buffer, the classifier decides what action to take. In the most simple transition-based parser that is the arc-standard parser [Nivre, 2003], the classifier can choose from amongst three actions which are left arc, right arc and shift. The shift action moves a word from the input buffer to the top of the stack and the other two actions add a dependency relation between either the top of stack and the word below it or the other way around and the word which is dependent in the relation is removed from the stack. The classifier keeps choosing an action until all the words have a dependency relation amongst them, the input buffer is empty and a correct parse tree is generated. Since we make only one left to right pass over all the words in a sentence, the complexity of these parsers is linear in the length of the sentence.

The reason it is called a transition parser is because you transition from the initial state through a set of states to the final state where the input buffer is empty and a correct parse tree is generated. Getting good features to train the classifier to classify well is difficult but in today's age of neural networks, several parsers make use of this technology to obtain dense and good feature representation for transition-based parsers [Kiperwasser and Goldberg, 2016, Chen and Manning, 2014, Weiss et al., 2015].

Graph-based parser

The idea behind a graph-based parser is that given an input sentence, first a fully connected directed graph is created where the vertices of the graph are the words of the input sentence and the edges represent the potential dependency relations between the words. Each edge is given a score. Any spanning tree of this graph can represent a possible parse tree for the sentence and so the parser must decide on which parse tree is the correct parse tree. The parser does this by treating it as a *maximum spanning tree* problem where the parser needs to find the dependency tree having the highest score [McDonald et al., 2005]. The score of a dependency tree is the sum of the scores of the dependency relations in the tree. Thus a graph-based parser tries to search through the space of available parse trees for the correct dependency tree.

A parser needs to learn by training on data from treebanks on how to score these edges. A higher score needs to be associated for correct dependency relations than incorrect ones. Similar to how transition-based parsers these days have adapted neural networks, graph-based parsers also make use of neural networks to obtain better feature representations and better scoring mechanism [Kiperwasser and Goldberg, 2016, Ji et al., 2019, Dozat and Manning, 2016]. In our work, we make use of a neural graph-based parser called the BIST parser [Kiperwasser and Goldberg, 2016] to further compliment our findings from our main choice of parser.

Sequence labelling parser

In recent years, quite a few studies have reduced constituency parsing or dependency parsing to a sequence labelling problem [johanka Spoustová and Spousta, 2010, Li et al., 2018, Strzyz et al., 2019c, Gómez-Rodríguez and Vilares, 2018, Vilares et al., 2019, Strzyz et al., 2019a]. The basic principle is to encode the dependency relations into a set of labels and then for each word we need to predict the correct label. An advantage of doing such a thing is that it is no longer required to have an explicit parsing structure or parsing algorithm [Strzyz et al., 2019c]. Although these parsers suffer a bit in terms of accuracy, they make up for it by being extremely fast. In our work, we choose to use a sequence labelling parser as our main parser of choice. A study by Vilares et al. [2019] found that even for small amounts of training data, a sequence labelling parser works reasonably well which works in our favour since our dataset is quite small. Another study by Li et al. [2018] showed that their parser achieved results which were at par with traditional transition or graph-based parsers. The work done by Strzyz et al. [2019c] is perhaps one of the most influential one in making us choose to cast our problem as sequence labelling. Their dependency parser achieves competitive results with the more standard neural parsers and it also provides a good trade-off between speed and accuracy of the parser.

Taking all these studies into consideration, it seemed like a good idea to cast our problem as sequence labelling since although we need to sacrifice some of the accuracy, the parser still has results that are at par with standard neural parsers and the gains in speed that it provides us are well worth this sacrifice since it would let us focus on how best the eye-tracking features can help in improving the parser by letting us run more experiments in the same time[1].

1.1.3 Evaluation

A parser is usually evaluated on the basis of two metrics, namely the *Labelled Attachment Score* (LAS) and the *Unlabelled Attachment Score* (UAS). The LAS is concerned with the number of words that are assigned a correct head as well as the correct dependency relation whereas the UAS is just concerned with the words that are assigned a correct head.

[1] Also achieving state-of-the-art results is not the goal of this thesis as mentioned before.

1.2 Eye-tracking

Eye-tracking information has been used over several years to try to understand cognitive processes. It is especially informative when it comes to the process of reading. Intuitively, one can understand how information about the piece of text that is being focused on and how long it takes to read a piece of text would help in understanding how text is comprehended while being read. There are studies which seem to indicate that when a person reads a sentence they analyze it word by word and build a syntactic structure associated with the sentence in an incremental manner [Frazier and Rayner, 1982, Just and Carpenter, 1980]. This section has been influenced by the work of Rayner [1998] and Gaskell et al. [2012] in particular.

1.2.1 What happens when we read?

When we read some text, the eye moves in short, quick jumps known as *saccades* [Rayner, 1998]. These saccades usually last between 20-40 milli-seconds (ms). In between saccades, the eye is stationary and this is known as a *fixation*. The minimum duration of a fixation is around 50 ms, the maximum is around 500 ms and usually it lasts between 200-250 ms. One can glean information about what they see only during fixations and not during saccades. This is because during a saccade, the eye moves so fast that only a blur can be seen [Uttal and Smith, 1968]. The reason why one doesn't perceive this blur during a saccade is because our mind is still processing the information that was available before and after a saccade [Brooks et al., 1981, Chekaluk and Llewellyn, 1990].

The visual field of what an eye can see can be divided into three regions the first of which is the *foveal* region which is the part of the text which is in 2° of the visual angle on the left and right of the fixation point. Beyond this region is the *parafoveal* region which extends upto 5° to the left and right of the fixation point. Beyond the foveal region, the sharpness of what we see deteriorates rapidly and in the parafoveal region we can only glean some information about the identity of a letter. Beyond the parafoveal region extends the *peripheral* region from which one can understand only the most general shape of the text. The entire point of a saccade is to bring a new piece of the text in the foveal region because a reader simply cannot understand the information from the parafoveal and peripheral region well enough [Rayner and Bertera, 1979, Rayner et al., 1981].

With a typical saccade while reading English, the eye moves about 7-9 spaces but there can be a lot of saccades which don't stick to this and can move the eye by even 15 spaces. McConkie and Rayner [1975] and Rayner and Bertera [1979] show that while reading English, the space around the fixation point from which one can actually glean information is not symmetric. It rather extends 14-15 spaces to the right of the fixation point and only 3-4 spaces to the left of the fixation point. The region in which a reader can actually understand the words also known as the *perceptual span* is dependent on the difficulty of the text and is usually 7-8 letter spaces [Rayner, 1998]. A study by Pollatsek et al. [1993] found that a reader doesn't obtain any information from the lines that are below the line that is fixated. These insights tell us that a reader only understands the word that is fixated along with the next word and studies also show that the

second word is read 30-40 ms faster if it is visible as part of the previous word's fixation [Hyönä et al., 2004, Rayner, 1998]. For a skilled English reader, 90% of the saccades are forward saccades while the rest of the times the eye moves backward to resolve any issues in understanding the text or to correct the forward saccade in case it moves the eye too far ahead. Such a backward saccade is called a *regression* [Rayner, 1998].

1.2.2 Factors affecting eye-movements while reading

It seems to be the case that the number of times a word is fixated corresponds to the length of that word and that very short words are quite often entirely skipped [Brysbaert and Vitu, 1998, Rayner and McConkie, 1976]. A possible explanation for skipping short words is that information regarding them is already obtained from the previous fixation [Inhoff and Rayner, 1986, Gaskell et al., 2012]. Studies also found that words are skipped over if they can be easily predicted from context and this seems to make sense on an intuitive level [Gautier et al., 2000, Rayner and Well, 1996]. A study by Rayner et al. [1996] also found that words that appear quite frequently in text are often skipped and in line with this it was also found that function words like prepositions or determiners are skipped more than half the time [Carpenter and Just, 1983]. The frequency of the word also affects the duration of its fixation [Liversedge et al., 2004, Rayner et al., 2006, 2003] and this seems to indicate that fixation durations are more indicative of higher level linguistic processing rather than low level perceptual processing.

Word frequency affects the total duration of fixation for that word as well as the first fixation time for the following word [Rayner and Duffy, 1986, Inhoff and Rayner, 1986]. Rayner and Duffy [1986] also observed that there was an increase in the time required to read a word which followed a low frequency word and theorized that this might be due to a difficulty in incorporating the low frequency word with the previous context. Some factors that affect eye-movements but are independent of the affect of word frequency are familiarity with a word and the age of acquisition of the word. The more familiar with a word the reader is, the faster it is processed by them and less familiar words have longer first fixation durations as compared to familiar words [Chaffin et al., 2001, Juhasz and Rayner, 2003, Williams and Morris, 2004]. Also words that are acquired early in life by a reader are processed much faster than words they acquire in the later stages of their life [Juhasz and Rayner, 2003, Juhasz, 2005, Juhasz and Rayner, 2006]. It was also found that a word was decomposed into its morphemes while being read and that the frequency of the constituent morphemes also affects the fixation duration of the word for English and Finnish languages [Andrews et al., 2004, Pollatsek and Hyönä, 2005, Pollatsek et al., 2000, Hyönä and Pollatsek, 1998, Juhasz et al., 2003].

The ambiguous meanings of a word also affect the eye-movement patterns [Rayner and Duffy, 1986, Duffy et al., 1988, Rayner and Frazier, 1989, Sereno et al., 2006]. Studies found that when a word has multiple meanings and each of these meanings have more or less the same frequency, then there is an increase in the fixation duration for the word as compared to an unambiguous word having the same word frequency and length. However, if one of the meanings is more dominant then there is no increase in the fixation duration. Then again, on

reading further if it turns out that the subordinate meaning is indeed the correct meaning rather than the dominant one, then there are more regressions and longer fixations to correct the mistake. It was also observed that if there is some information which disambiguates a following ambiguous word whose multiple meanings have equal frequencies, then there is no increase in the fixation duration. However, if it turns out that the disambiguating information points towards the subordinate meaning of the word, then there is an increase in the overall reading time of the word. Frazier and Rayner [1987] found that when there is ambiguity regarding the syntactic category of a word and the categories are equiprobable to appear in the given context then there is an increase in the reading time only later down the line when we come across the information which disambiguates the category.

Studies also found that whenever the predictability of a word based on the past context increases, its reading time decreases and some words that are highly predictable from the past context are altogether skipped [Ehrlich and Rayner, 1981, Rayner and Well, 1996]. Morris [1994] found that the fixation duration on a word decreases if it is semantically associated with any prior word.

Most studies involved in trying to analyze the effect of syntax on eye movements focus on the region of the text which resolves some syntactical ambiguity with the idea being that at the point the reader reads this disambiguating text, there will be regressions or longer fixation durations which means that the initial syntactical analysis that the reader had constructed is wrong [Frazier and Rayner, 1982]. Since one can easily tell from eye movement recordings whenever there is an abrupt change while reading for instance by sudden regressions or longer gaze durations whenever an initial syntactic analysis of a reader is refuted, several studies tried to test linguistically motivated theories regarding the parsing strategies employed by a person when he reads [Gaskell et al., 2012]. Frazier and Rayner [1982] and Meseguer et al. [2002] found that a reader is innately aware of the point in text where their incorrect analysis differs from the correct analysis as the reader's eyes move back to this point after a regression. A few studies also show that there is an increase in the fixation duration for a word that ends a clause or a sentence and this can be attributed to the time required for integrative processing at sentence boundary [Rayner et al., 2000, Just and Carpenter, 1980].

1.2.3 Overview of eye-tracking in NLP

In this section we describe some of the studies in the field of NLP which make use of eye-tracking data. Barrett et al. [2016a] use eye-tracking data to improve a POS tagger and found significant improvements in the results. A very interesting study by Barrett et al. [2016b] found that the correlation between gaze features and POS tags can be transferred across languages and they use English gaze features to improve a French POS tagger. Barrett and Søgaard [2015a] found that eye-tracking information can be used to predict the syntactic categories of words and Barrett and Søgaard [2015b] used gaze features to predict the grammatical function of a word. The study done by Barrett et al. [2018] leveraged eye-tracking information to find out human attention and used it to regularize the attention function used in an RNN and found improvements for a range of NLP tasks like sentiment analysis, abusive language detection, etc.

Bingel et al. [2018] use eye-tracking information to predict reading errors of children that have some reading disability. Metrics for evaluating the quality of machine translation output derived from eye-tracking data were found to be better than the automatic metrics in use [Klerke et al., 2015]. The idea that eye-tracking data could be used to evaluate machine translation output seems quite reasonable as bad translations would result in longer fixation durations and more regressions by the reader which can be picked up from the data. Klerke et al. [2016] in their work use gaze data in a multi-task learning setup to improve sentence compression. Søgaard [2016] and Hollenstein et al. [2019] couple fMRI data along with eye-tracking data to evaluate the quality of word embeddings. Hollenstein and Zhang [2019] leveraged eye-tracking data for improving named entity recognition. An important feature of their study was to leverage type-aggregated gaze features to eliminate the need for recording eye-tracking data at test time and also make the features useful for cross-domain settings.

A couple of studies have also experimented with some sort of parsing of text using eye-tracking data. Strzyz et al. [2019b] leverage gaze data by learning eye-movement features as an auxiliary task in a multi-task learning (MTL) setup where both dependency parsing and gaze prediction are addressed as sequence labelling. Their experiments resulted in small positive improvements to dependency parsing, however they did not measure the statistical significance of their results. Lopopolo et al. [2019] go about dependency parsing the other way around by showing that there is a relation between regressions and the syntactic structure of sentences. They tested if the path of regressions from a word to an earlier word coincide – at least partially – with the edges of dependency relations between these words by using dependency parsing features to predict eye-regressions during training. One of their important findings indicates that eye regressions are involved predominantly in dependency parsing at the local level (vast majority being shorter than three words with a predominance of one position backwards), rather than at long distance. Cheri et al. [2016] utilize the eye-movements of several annotators resolving coreference to improve automatic coreference resolution.

Mathias et al. [2018] rate the quality of a piece of text by using eye-tracking data and Mishra et al. [2017] try to quantify the effort needed in reading a piece of text by measuring the complexity of the scanpath of various readers. A few studies also found that eye-tracking data can be used to improve sentiment analysis as well as sarcasm detection [Mishra et al., 2016a,b,c].

1.2.4 The state of eye-tracking technology

One of the primary reasons for a spurt of studies in NLP with eye-tracking data is that although initially eye-trackers were considered an expensive and complex piece of equipment available only in highly funded and well known labs, in the past decade or so there have been massive improvements to hardware infrastructure resulting in significantly cheaper eye trackers and an increased availability of them. Eye-trackers are no longer considered as part of the domain of well funded labs and can be afforded and used by the common man as well.

Even if we take into consideration that any commercial eye-tracker is still a bit expensive for an individual to afford, a study by San Agustin et al. [2009]

found that a low cost web-cam based eye-tracker can still be almost as efficient as commercial eye-trackers for certain tasks. Taking this a step further towards making eye-tracking technology pervasive, Krafka et al. [2016] built an eye-tracking software that can work on hardware such as smart phones or tablets without any additional sensors or devices by making use of the advanced hardware and the massively improved cameras with high definition available in these devices. A system called the *EyeTab* was developed by Wood and Bulling [2014] which is a model based approach for determining gaze information which can run on any tablet without extra sensors or devices. Lukander et al. [2013] developed an affordable, 3-D printed, wearable mobile gaze tracker which when developed cost approximately 350 euros.

San Agustin et al. [2010] developed the ITU Gaze Tracker which was a low-cost eye-tracking system based on a webcam that is mounted close to the user's eye. The developers of this system soon established their company *"The Eye Tribe"*[2] with a focus on providing low cost eye-tracking and providing eye control technology for mass market consumer devices.

Recently, even gaming consoles like Sony's Playstation is making use of eye-tracking technology for their VR headsets and motion capture systems and thus researchers could make use of these systems for their own purposes. These advancements bring with them a promise that eye-tracking technology will be pervasive in the coming years and thus can be leveraged for NLP tasks as an alternative source of information.

1.2.5 Eye-tracking corpora

The eye-tracking corpora mentioned in this section are just for an indicative idea and are some of the most well known and frequently used in NLP. There are quite a few other corpora also available.

The Dundee corpus [Kennedy et al., 2003] is one of the oldest and most used corpus in eye-tracking studies. It contains eye movement recordings for English and French text and employed a *Dr. Bouis Oculometer Eyetracker* for recording the measurements. Participants were recorded while reading newspaper articles from *The Independent* for English and *Le Monde* for French. The participants included 10 native speakers of each language reading 20 newspaper articles from either newspaper source. The English corpus contains 51,502 tokens [3] and 9,776 types in 2,368 sentences. Participants read the text five lines at a time.

The Ghent eye-tracking corpus (GECO) [Cop et al., 2017] is one of the largest eye-tracking corpus by token count and it is much more recent than the Dundee corpus. There were 17 participants who read the entire novel *The Mysterious Affair at Styles* by Agatha Christie and their eye movements were recorded with the help of an *Eye-Link 1000* tracker. It is an English-Dutch bilingual corpus with the English part of the corpus containing 54,364 tokens in 5,031 sentences. Participants read the text one paragraph at a time.

The Zurich Cognitive Language Processing (ZuCo) corpus [Hollenstein et al., 2018] has a combination of eye-tracking data as well as electroencephalography (EEG) recordings. The participants for the corpus were 12 adults who were native

[2] `https://theeyetribe.com`

[3] Punctuations and contracted words are considered as part of preceeding token.

speakers of English and their eye movements were recorded with the help of an *Eye-Link 1000* tracker. The corpus contains 21,269 words in 1,100 sentences and was presented to the participants one sentence at a time on a screen at the same position. There were two normal reading tasks and one task-specific reading and the eye-tracking data consists of about approximately 154,173 fixations. The sentences in the corpus are from movie reviews from the *Stanford Sentiment Treebank* and biographical sentences about notable people from the *Wikipedia relation extraction corpus*.

The Provo Corpus [Luke and Christianson, 2018] contains around 2,700 English words read in 55 short paragraphs by 89 native speaking participants. It comes with predictability norms and has 134 sentences in total.

2. A Primer on Neural Networks

2.1 Neural networks

The success of convolutional neural networks in image recognition tasks led to a renewed interest by researchers in deep learning and people in the NLP community also turned their attention towards incorporating neural models for several NLP tasks. Feed forward neural networks that are fully connected are non-linear models that can be used in place of a linear model. Such a replacement led to a number of studies [Chen and Manning, 2014, Weiss et al., 2015, Pei et al., 2015, Durrett and Klein, 2015] obtaining better results for syntactic parsing.

For NLP, the input features that are usually used are words, POS tags, morphological features, etc. Earlier, the way these features were represented was using the one-hot encoded representation but this lead to a very sparse representation where each feature had its own dimension, the dimensionality of the vector was the same as number of distinct features, and the features were completely independent from one another. To avoid this, researchers tried to obtain a dense representation by projecting the features into a fixed dimension vector space. These dense features share similarities unlike the sparse representation (for e.g. the dense representation for the words 'king' and 'queen' are more similar than for say the words 'king' and 'tree'). Mikolov et al. [2013a] describe two methods for obtaining dense representations namely: the continuous bag-of-words model and the continuous skip-gram model.

There are generally two approaches for obtaining dense representations, that is, task-specific embeddings and pre-trained embeddings. In task-specific embeddings, the idea is to optimize the performance of the embeddings on the task they are being trained on. A potential disadvantage of this method is that sometimes the data required to train these embeddings properly might be too large to be available for some tasks. Pre-trained embeddings are trained independently of the task in which they are used. The word-2-vec embeddings by Mikolov et al. [2013b] are one of the most famous pre-trained embeddings that have been used in several tasks. Nowadays, BERT embeddings [Devlin et al., 2019] which are contextualized word embeddings are quite the rage in NLP. Usually these embeddings are trained for some kind of language modelling tasks as that doesn't require any particular annotation and thus any large corpus would suffice for training these embeddings.

In NLP, we usually work with sequences and we would like to preserve the structural information about the sequence and recurrent neural networks allow us to do exactly that [Elman, 1990]. The use of recurrent networks in many NLP tasks like dependency parsing, sentiment analysis, language modelling, machine translation, etc. have produced excellent results [Ballesteros et al., 2015, Dozat and Manning, 2016, Kiperwasser and Goldberg, 2016, Wang et al., 2015, Auli and Gao, 2014, Sutskever et al., 2014]. A recurrent neural network (RNN) is made up of cells and each cell feeds its output to the next cell which the next cell combines with its own independent input to give its output to the next cell and so on. Providing this past output to the next cell lets us condition our model on the entire past history.

The issue with the simple RNN model was what is called the vanishing gradient problem where the gradient becomes extremely small rather rapidly during the back-propagation process and doesn't reach the initial cells making it hard to capture long range dependencies. Hochreiter and Schmidhuber [1997] proposed the Long Short-Term Memory (LSTM) architecture to solve the issue of vanishing gradients. The LSTM architecture is responsible for producing state-of-the-art results in several sequence modelling tasks and it is also the architecture we use for our experiments.

2.2 Sequence labelling

Sequence labelling is a type of NLP task that involves assigning a categorical label to each token/word in a given input sequence. The simplest sequence labellers simply assign each token a label independent of the surrounding tokens but they can also be conditioned to take into account the labels assigned to the past tokens or just the surrounding tokens.

There are usually two kinds of sequence labelling:

- Token labelling: Here each token in an input sequence is individually assigned a label. Common application of this is for POS tagging. Figure 2.1 shows an example of token labelling.
- Segmentation labelling / Span labelling: Here segments or group of tokens are assigned one label. Common application of this is for named entity recognition (NER). The spans are labelled with a **BIO** tag representing the **B**eginning, **I**nner and **O**utside of entities. Figure 2.2 depicts an example of segmentation labelling.

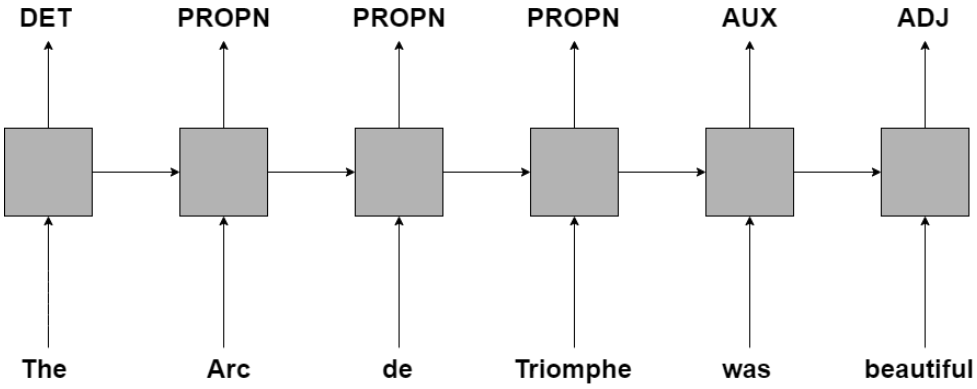

Figure 2.1: A basic example of POS tagging as sequence labelling

2.3 Multi-task learning

Caruana [1997] defines multi-task learning (MTL) as follows: "It is an approach for inductive transfer that improves generalization by using the domain informa-

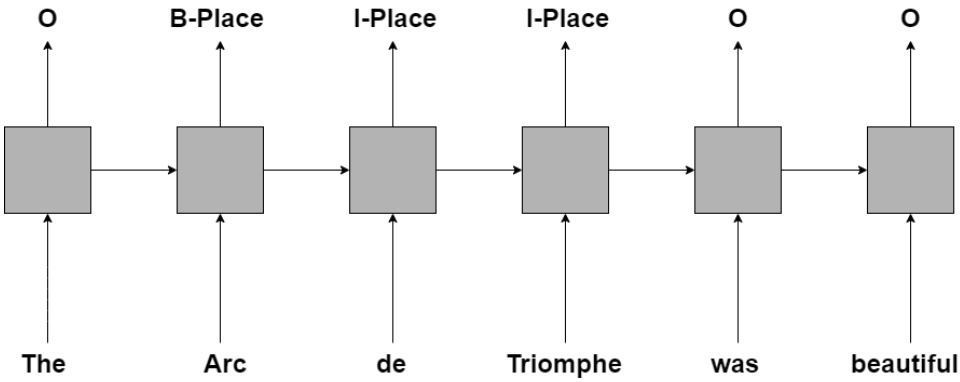

Figure 2.2: A basic example of NER as segmentation labelling

tion contained in the training signals of related tasks as an inductive bias". The idea behind multi-task learning is to use the knowledge gained from other tasks to improve your primary tasks. The reason it works is because any knowledge gained from the other auxiliary tasks will always help in some way or the other with the primary task. Also when you use the same network to solve multiple tasks, the network is forced to learn a more general representation that is more suitable to all the tasks as opposed to a more concrete representation for a fixed task. Thus in some sense it can also be viewed as some kind of a regularization factor.

Figure 2.3 shows the first kind of approach for MTL which is known as the hard parameter sharing. Here the initial layers are shared between the tasks and then for each task, there are a few task-specific layers. Figure 2.4 shows the general structure for soft parameter sharing which is the second approach for MTL. Here each task has its own individual network but the distance between the model parameters for each network is regularized to try to make them more similar,

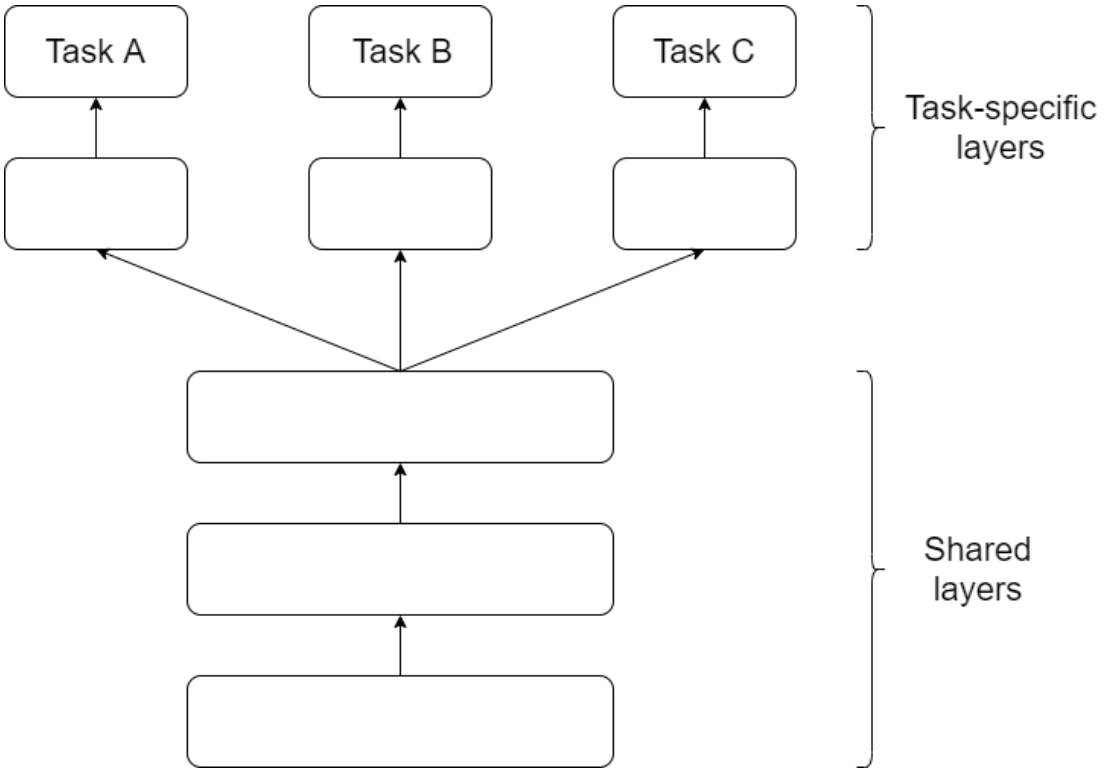

Figure 2.3: Hard parameter sharing for multi-task learning; source: Ruder [2017]

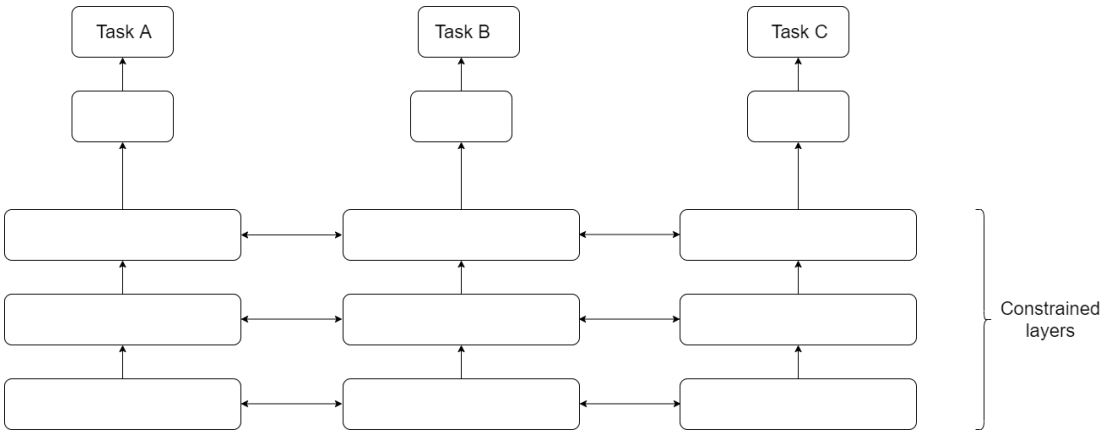

Figure 2.4: Soft parameter sharing for multi-task learning; source: Ruder [2017]

3. Parser Descriptions

In this chapter, we describe the two parsers we use in our work. Our primary parser is a modification of the system created by Strzyz et al. [2019b] who in turn adapted the *NCRF++* system [Yang and Zhang, 2018] which is an open source neural sequence labelling toolkit. Our secondary parser is a more traditional graph-based parser known as the *BIST* parser from Kiperwasser and Goldberg [2016].

3.1 Dependency parsing as sequence labelling

The first step while casting dependency parsing as a sequence labelling problem is to convert the dependency tree representation into a set of labels. Our data is in the form of the CoNLL-X format with additional columns containing the gaze features. Listing 3.1 shows an example of data in this format.

```
6   on    _   ADP  IN   _   10  case    _   _   231.0   1.0  121.7
7   the   _   DET  DT   _   10  det     _   _   208.75  0.0  145.5
8   chair     _    NOUN     NN   _   3   obl   _   _   220.2   1.0  23.1
```

Listing 3.1: Example of data in CoNLL-X format with additional columns for gaze features

Now, we follow the procedure by Strzyz et al. [2019c] for casting dependency parsing as sequence labelling and they state that for each word w_i in a sentence s, associate it a with a node ranging from $\{0, 1, \ldots, n\}$, where n is the total number of words in the sentence and 0 is considered as the dummy root of the sentence. Then a parser will find dependency relations between these words encoded in the form of a triple (h, d, l) where h is the head, d is the dependent and l is the dependency relation label. The dependency graph that is formed should be acyclic where each node has only one head except the root node so that the graph is a directed tree rooted at node 0.

To encode a dependency tree as labels for sequence labelling, Strzyz et al. [2019c] found that it is sufficient to just encode a word's head and dependency relation associated with it for all the words in a sentence. So following the strategy by Strzyz et al. [2019c] for relative positional encoding, we provide each word with a label that contains its dependency relation label as well as the relative position of its head. The relative position of the head is an encoding in the form of a tuple p_i, o_i of a POS tag p_i and a positive or negative number o_i. If the number is positive then it means the head of the word is the o_ith closest word to its right having the POS tag p_i and if the number is negative then the head of the word is the o_ith closest word to its left having the POS tag p_i. For example, (N, 1) would mean "the first noun on the right" of the word is its head. Figure 3.1 shows an example of an encoded dependency tree.

The generated labels are also conditioned on the chosen multi-task learning (MTL) setup. In our case, we make use of hard parameter sharing for any chosen MTL setup. We make use of two setups from amongst the setups described by [Strzyz et al., 2019a], namely:

- single: The label will be treated as a single task where its components are

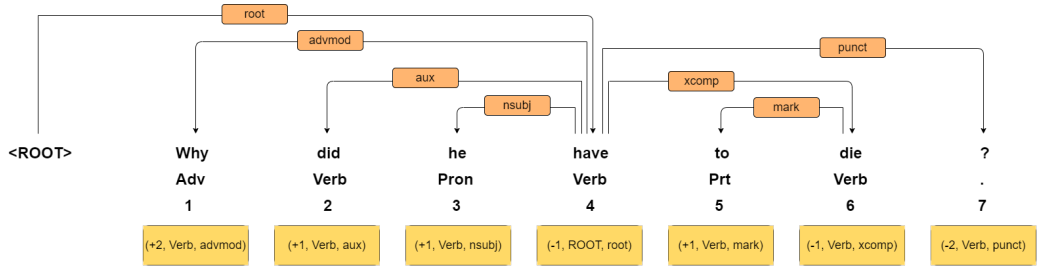

Figure 3.1: Example of an encoded dependency tree

separated by the symbol "@" [Strzyz et al., 2019a]. An example sentence with this setup is provided below.

Why	Adv	+2@advmod@Verb
did	Verb	+1@aux@Verb
he	Pron	+1@nsubj@Verb
have	Verb	-1@root@ROOT
to	Prt	+1@mark@Verb
die	Verb	-1@xcomp@Verb
?	.	-2@punct@Verb

- combined: In this setup, information about the relative position and word's head (e.g. +1@Verb) is combined into one task and dependency relation is considered as the second task [Strzyz et al., 2019a]. Each component separated by "{}" will be treated as a separate task. As per Strzyz et al. [2019a] this gives us the best performance. An example sentence with this setup is as below.

Why	Adv	+2@Verb{}advmod
did	Verb	+1@Verb{}aux
he	Pron	+1@Verb{}nsubj
have	Verb	-1@ROOT{}root
to	Prt	+1@Verb{}mark
die	Verb	-1@Verb{}xcomp
?	.	-2@Verb{}punct

3.1.1 Architecture

We use bi-directional long short term memory (bi-LSTM) networks [Hochreiter and Schmidhuber, 1997, Schuster and Paliwal, 1997] for the sequence labelling model following the work of Strzyz et al. [2019b]. By adding a feed-forward network which takes as input the output of the bi-LSTM, we predict the label for each token. A separate feed-forward network is used for each task according to the MTL setup.

Figure 3.2 depicts the broad architecture of our dependency parser which follows the architecture of Strzyz et al. [2019b]. For the inputs, we use word forms, characters in the word and POS tags which are passed through an embedding layer which provides us with their dense representation. The gaze features are

also a part of the input, however since they are already real-valued, they are not 'embedded'. One of the changes we make to the parser by Strzyz et al. [2019b] is to concatenate the word,character and POS embeddings with the corresponding gaze features of the word and pass it through a bi-LSTM, which generates vectors that take context into account. The bi-LSTM can be seen as a concatenation of two LSTMs, one which processes the input from left-to-right and the other processes the input from right-to-left. We stack two layers of bi-LSTMs before decoding the output similar to Strzyz et al. [2019b].

The hidden state of every LSTM cell in the final LSTM layer is then passed to two separate feed-forward layers (followed by a softmax) to predict the corresponding label for each of the tasks [Strzyz et al., 2019b]. For each cell (each of which is meant to represent a token), one of the feed-forward layers is meant to predict the index of the head term, whilst the other is meant to predict the dependency relation label [Strzyz et al., 2019b]. For the combined MTL setup, the cross-entropy loss is computed as:

$$\mathcal{L} = \mathcal{L}_{(o,p)} + \mathcal{L}_d$$

3.1.2 Hyperparameters

Parameter	Value
Word embedding dim.	100
POS embedding dim.	25
Optimizer	SGD
Epochs	150
Batch size	8
LSTM hidden state dim.	800
LSTM layers	2
Learning rate	0.02
Learning rate decay	0.05
Momentum	0.9
Dropout	0.5
Char. embedding dim.	30
Char. LSTM dim.	50
Char dropout	0.5

Table 3.1: Hyperparameters for the parser

We keep most of the parameters used by Strzyz et al. [2019b] in their paper as the goal of this thesis is more of an exploratory nature as opposed to achieving state-of-the-art results in dependency parsing. Our hyperparameters are described in Table 3.1. We trained each model upto 150 epochs and kept the model with the highest score on the development set. During multitask learning, for the combined setup we weigh both the tasks as 1.0 since both are equally important and are our main tasks.

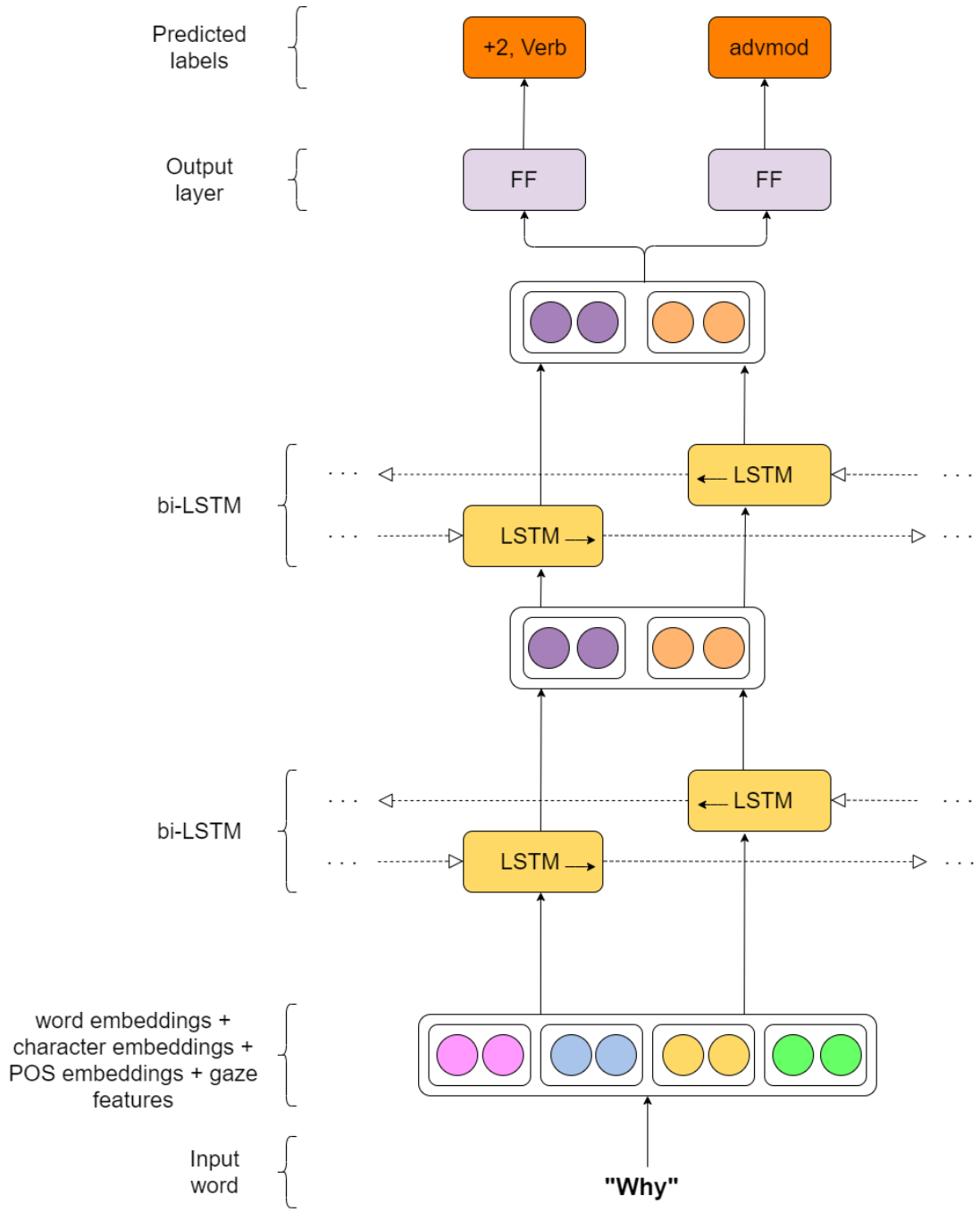

Figure 3.2: Architecture of sequence labelling parser in combined MTL setup adapted from Strzyz et al. [2019b]

3.1.3 Evaluation metrics

For all our experiments, we evaluate the sequence labelling parser with respect to the Unlabelled and Labelled Attachment Score (UAS and LAS) excluding punctuations. We exclude the punctuations following the previous work done in this area [Strzyz et al., 2019b].

3.1.4 Implementation details

The source code for our project can be found online[1]. This repository is a clone of the repository by Strzyz et al. [2019b] with a few slight modifications to suit our needs. In the neural model while concatenating the word, character and POS tag embeddings for the word representation, we also concatenate the gaze features at this step. Depending on the parser setup, sometimes we exclude the POS tag embeddings or the word and character embeddings. We also add the gaze features while encoding the dependency tree to sequence labels and added a proper decode mode to the parser when it loads an already trained model to parse new sentences. In addition to these changes, there are also some simple data-processing scripts for normalizing and averaging the data.

3.2 A Brief Overview of the BIST Parser

To provide a better perspective about incorporating gaze features with dependency parsing, we also try out a more standard or traditional parser which is commonly known as the *BIST* parser from Kiperwasser and Goldberg [2016]. Kiperwasser and Goldberg [2016] employ their approach for both a transition-based parser and a graph-based parser, however we only make use of the graph-based parser in our work. The main problem that Kiperwasser and Goldberg [2016] focused on was how to obtain good features for the central statistical model. They suggest using bi-LSTMs for getting the feature representations since they are good at capturing the meaning of an element in a sequence along with the context. An important recommendation from them is to train the bi-LSTM together with the rest of the parser to get good feature representations so the structured prediction model is jointly trained with the bi-LSTM encoder with the errors being propagated up to the bi-LSTM feature encoder. The model that they use for the graph-based parser is the simple first-order arc-factored model [Mcdonald, 2006].

So according to Kiperwasser and Goldberg [2016], if for a sentence s comprising of n words $w_1, \ldots, w_n$ where we also have the corresponding POS tag p_i for each word w_i, we get the embeddings for the words and POS tags denoted by $e(w_i)$ and $e(p_i)$ and then create a sequence of input vectors $x_{1:n}$ where each vector x_i is a concatenation of the word embedding and POS tag embedding and $\circ$ is the concatenation operator:

$$x_i = e(w_i) \circ e(p_i)$$

For our work, we only make one addition to the parser. We use seventeen different gaze features denoted by $g_1, \ldots, g_{17}$ and we concatenate each of these

[1] `https://github.com/balthamel/dep2label-up`

gaze features directly to the word and POS tag embeddings. So our input vector would look like:

$$x_i = e(w_i) \circ e(p_i) \circ g_1 \circ \ldots \circ g_{17}$$

To further get the contextualized embedding for each word (v_i), the bi-LSTM layer is used. Figure 3.3 shows the architecture of the graph-based BIST parser. The dependency relations are shown below the actual words and the multi-layer perceptron (MLP) scores every dependency arc of the sentence. In the end, all the arc scores are added to obtain the final score. For a sentence, all possible combination of the arcs are scored (n^2) and the tree having the highest score is then chosen.

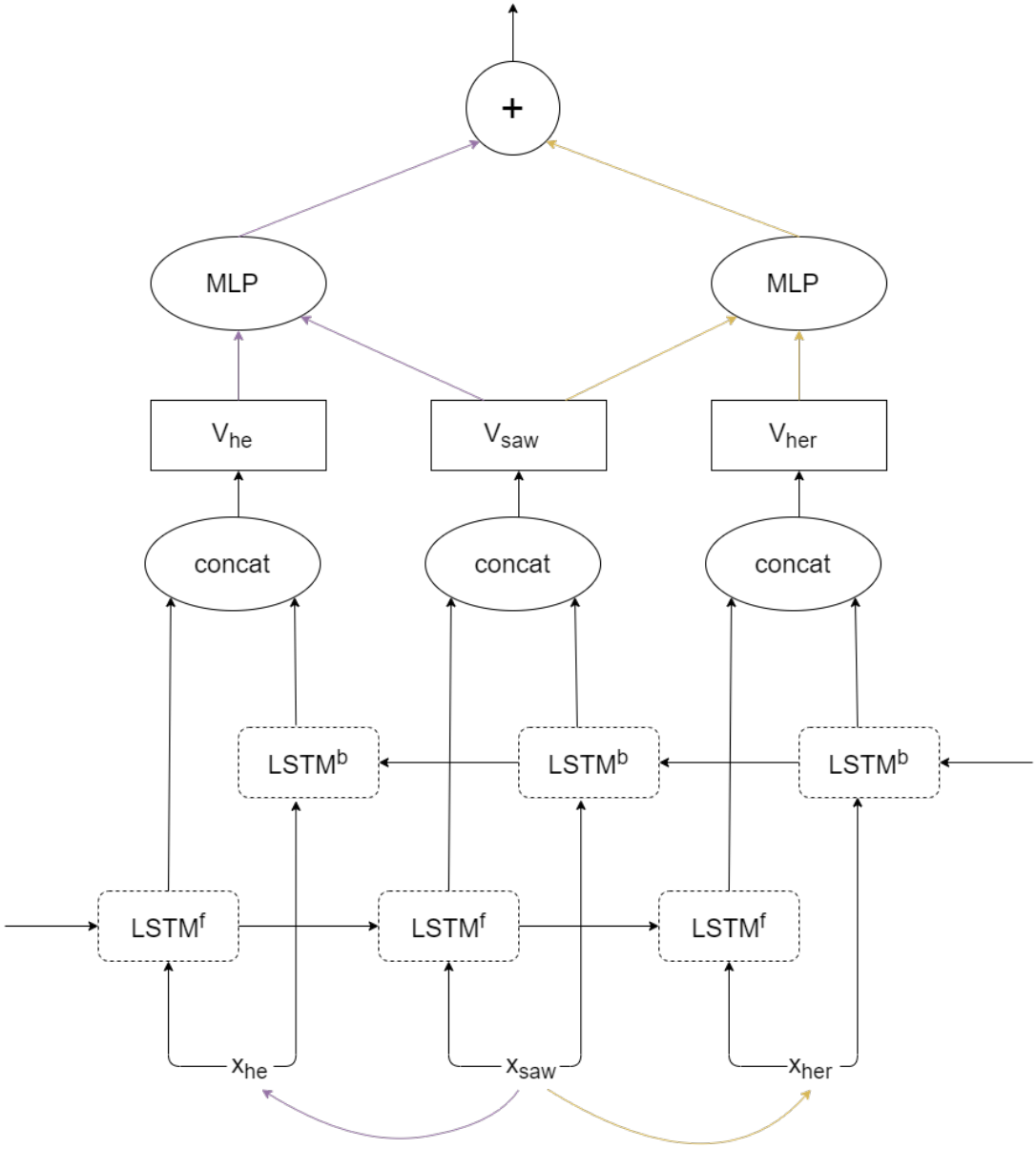

Figure 3.3: Architecture of the graph-based BIST parser; source: Kiperwasser and Goldberg [2016]

3.2.1 Implementation details

The source code for the graph-based BIST parser can be found online[2]. This repository is a fork of a repository which re-implements the original code of Kiperwasser and Goldberg [2016] in pytorch. As mentioned earlier, our only contribution to this code is to incorporate the various gaze features at the time of concatenating the other embeddings.

3.2.2 Hyperparameters

Parameter	Value
Word embedding dim.	100
POS embedding dim.	25
Optimizer	Adam
Epochs	50
LSTM hidden state dim.	800
LSTM layers	2
Learning rate	0.1
MLP hidden state dim.	100

Table 3.2: Hyperparameters for the BIST graph-based parser

Our hyperparameters are described in Table 3.1. We kept most of the original parameters and only changed the dimension of the LSTM hidden state to 800 because the original parameter was giving poorer results. We trained each model upto 50 epochs and saved the model after every epoch.

3.2.3 Evaluation metrics

For all our experiments, we evaluate the parser with respect to the Unlabelled and Labelled Attachment Score (UAS and LAS). We also include punctuations in the evaluation this time since for our previous parser we excluded them and we hope to demonstrate the overall effect of gaze features on dependency parsing by covering all possible cases.

[2] `https://github.com/balthamel/bist-parser`

4. Using Gaze Features for Dependency Parsing

In this chapter, we describe the dataset that we use along with a description of the gaze features that we intend to make use of. We then describe a set of experiments with these gaze features and report the UAS and LAS scores on the development set and analyze the result of all the scenarios.

4.1 Data

For all our experiments, we make use of the Dundee Corpus [Kennedy et al., 2003] which was referenced in Section 1.2.5. The reason we chose this corpus was because it is the only available eye-tracking corpus having a Universal Dependencies (UD) style syntactic annotation layer on top of the English side of the corpus as described by Barrett et al. [2015] in their work on the Dundee Treebank. A point to note is that, the original Dundee corpus was tokenized by white-space such that punctuation and word contractions are considered as a single token e.g. "What?" and "wasn't" are treated as a single token respectively. However, for the dependency annotation, Barrett et al. [2015] split the tokens into "What", "?", "was" and "n't". The eye-tracking measures were simply duplicated such that "What" and "?" have the same eye-tracking features. There are also cases in the data wherein a reader does not fixate upon a particular word and thus the *first fixation duration* and other such values are empty for that word in the dataset. We consider all these empty values to be 0 when we process the dataset. We only use the English part of the Dundee Corpus for our experiments. The first set of experiments are carried out on the data from only one of the ten readers which is identified in the corpus by the label ***"sa"***. For the second set of experiments we average out the data over all ten readers.

The original Dundee Corpus [Kennedy et al., 2003] contains raw eye-movement recordings. These raw gaze trajectories have been pre-processed by Maria Barrett and Sigrid Klerke to extract word based eye-tracking measures. We use these extracted gaze features in our experiments. We choose 17 different gaze features spread out across four distinct groups similar to Barrett et al. [2016a], Hollenstein and Zhang [2019]. These groups define the different stages of the reading process and are as follows:

- ***Basic***: These features capture characteristics on a word-level, e.g. the number of all fixations on a word or the probability that a word will be fixated.
- ***Early***: The features capture lexical access and early syntactic processing and are based on the first time a word is fixated.
- ***Late***: These features reflect the late syntactic processing and general disambiguation. The features for words which were fixated more than once are more significant here.

- ***Context***: These features capture the gaze measures of the surrounding tokens.

Table 4.1 provides a description of all the gaze features used in our experiments.

Basic	
n fixations	total number of fixations on a word w
fixation probability	the probability that a word w will be fixated
mean fixation duration	mean of all fixation durations for a word w
total fixation duration	sum of all fixation durations for a word w
Early	
first fixation duration	duration of the first fixation on a word w
first pass duration	sum of all fixation durations when it is first visited
Late	
n re-fixations	number of times a word w is fixated (after the first fixation)
re-read probability	the probability of revisiting word w after making a first pass
Context	
total regression-from duration	combined duration of the regressions that began at word w
w-2 fixation probability	fixation probability of the word before the previous word
w-1 fixation probability	fixation probability of the previous word
w+1 fixation probability	fixation probability of the next word
w+2 fixation probability	fixation probability of the word after the next word
w-2 fixation duration	fixation duration of the word before the previous word
w-1 fixation duration	fixation duration of the previous word
w+1 fixation duration	fixation duration of the next word
w+2 fixation duration	fixation duration of the word after the next word

Table 4.1: Gaze features spread over four groups; source: Hollenstein and Zhang [2019]

4.2 Experiments with single reader's data

In this section, we describe our experiments with data taken from a single reader in the Dundee corpus. The reader who's data we work with has the label or the identifier **"sa"** in the corpus. The idea behind using the data from just a single user in the beginning was to get some initial findings and to verify that the model works with the provided gaze features as input. We also hoped that the findings observed from a single reader would translate over to the averaged data over all users to some extent and that these initial findings would give us a good idea of what to expect from the averaged data.

For all the experiments in this section unless otherwise mentioned, we use a 90-10-0 train-dev-test split with 2131 sentences in the training set and 237 sentences in the dev set as the Dundee corpus has no official train-dev-test split established. Although this kind of split without a test set was regrettable, these set of experiments were just to establish some initial findings to pave the way for experiments with averaged reader data. We also perform these set of experiments only with the sequence labelling parser.

4.2.1 Parsing with all features

The first thing we do is to run our parser with all the available features at hand i.e. word features, character features, POS tags and the gaze features. For the character embeddings we use a bi-LSTM to extract the features and concatenate it with the word embeddings, POS tag embeddings and the gaze features. This concatenated input is then passed to the sequence-labelling model.

For the gaze features, we normalize all the values across the data. For normalizing the values, we use `scikit-learn`'s normalization from it's pre-processing library with all the three norms i.e. `'l1'`, `'l2'` and `'max'` norm. The exact function we use is `sklearn.preprocessing.normalize()`. The choice of the norm seems to have a considerable effect on the results as is shown in the next couple of sections. If x is the vector of covariates of length n and assume the normalized vector is $y = x/z$ where z is the norm, then the three norms can be described as follows:

$$L1 : z = ||x||_1 = \sum_{i=1}^{n} |x_i|$$

$$L2 : z = ||x||_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$$

$$Max : z = max\, x_i$$

Other options for handling the data would be to scale the values between 0 and 1 as followed by some authors. We try this out in one of our experiments by using `scikit-learn`'s `MinMaxScaler`. Another approach would be to discretize the continuous feature values into quantiles and then compute the embedding for each discrete gaze feature where the dimension of the gaze feature embedding is the number of quantiles [Hollenstein and Zhang, 2019]. In one of our initial experiments, we discretized two of the gaze features with `scikit-learn`'s `KBinsDiscretizer` and computed it's embedding to concatenate with the other feature embeddings, however the results were poorer than those obtained by normalizing the values which dissuaded us from using this approach in any further experiments.

Using just the raw gaze feature values doesn't work well as we found out with one of our initial experiments. It degraded the performance of the parser severely compared to the baseline. Since the ranges for the different gaze features are considerably different, normalizing them makes a lot of sense.

We also have two variations in terms of the gaze features that we use. In the first case we use only two normalized gaze features out of the seventeen gaze features listed in Table 4.1 namely, *mean fixation duration* and *total fixation duration*. These features are amongst the ones that contributed the most to dependency parsing as per Strzyz et al. [2019b] and they are also a part of the *basic* feature group. In the second case we use all seventeen normalized gaze features and these features were picked keeping in mind the previous works in this area [Hollenstein and Zhang, 2019, Strzyz et al., 2019b]. The baseline system is a model where no gaze features were used.

The results for all the experiments are provided in Table 4.2 (best results are in bold). From the table, it is clear that the gaze features improve the UAS scores

Gaze features		dev set UAS	dev set LAS
	baseline	86.48	79.03
L1 norm	2 feats.	86.47	78.68
	17 feats.	**86.91**	**79.20**
L2 norm	2 feats.	86.43	78.87
	17 feats.	86.74	78.91
max norm	2 feats.	86.66	79.01
	17 feats.	86.76	78.91

Table 4.2: UAS and LAS scores for models trained on all features

more than the LAS scores. In fact there is an improvement over the baseline for the LAS score only when we use all 17 features with the L1 normalization norm. The results from this table would seem to indicate that using all 17 gaze features with the L1 norm is the most helpful.

4.2.2 Delexicalized parsing

The next setup that we try out is delexicalized parsing of the data. Delexicalized paarsing has been found quite useful in low resource and cross-lingual settings [Zeman and Resnik, 2008, Aufrant et al., 2016]. That being said, our interest in this setup is merely to explore if eye-tracking features are useful in any form of parsing. In this case we omit the word and character level features and only use the POS tag embeddings and the gaze features. Similar to Section 4.2.1, we use three different normalization norms and two sets of gaze features. The baseline system is a model where no gaze features were used and only the POS tag embeddings were fed as input.

Gaze features		dev set UAS	dev set LAS
	baseline	76.55	65.61
L1 norm	2 feats.	77.12	66.39
	17 feats.	76.77	65.98
L2 norm	2 feats.	77.08	66.70
	17 feats.	76.57	66.00
max norm	2 feats.	**77.14**	**66.95**
	17 feats.	77.14	66.82

Table 4.3: UAS and LAS scores for models trained on gaze and POS features

The results for all the experiments are provided in Table 4.3 (best results are in bold). The delexicalized parsing results seem to be much better than the previous set of results indicating that the gaze features definitely contribute towards delexicalized parsing. The gaze features seem to improve both the UAS

and LAS scores in each case with the margin of improvement being more for the LAS scores. The best improvement seems to be with just using 2 features and the max norm resulting in an improvement of +1.34 for LAS and +0.59 for UAS score.

4.2.3 Parsing without POS tags

In this setup we try out parsing without the POS tags by omitting the POS embeddings and only using the word embeddings, character level embeddings and the gaze features. Similar to Section 4.2.1, we use three different normalization norms and two sets of gaze features. The baseline system is a model where no gaze features were used and only the word and character embeddings were fed as input.

Gaze features		dev set UAS	dev set LAS
	baseline	22.49	**19.3**
L1 norm	2 feats.	**22.55**	19.16
	17 feats.	22.49	19.12
L2 norm	2 feats.	22.18	18.91
	17 feats.	22.28	19.12
max norm	2 feats.	22.12	19.03
	17 feats.	22.51	19.16

Table 4.4: UAS and LAS scores for models trained on gaze, word and character features

The results for all the experiments are provided in Table 4.4 (best results are in bold). The results in this table do not really tell us much as to how gaze features contribute towards parsing without POS tags. All the scores in general are extremely low as compared to the previous settings. In fact, the LAS scores seem to drop in each case after the addition of gaze features and there is only a marginal increase in the UAS scores in some cases which doesn't really tell us anything.

4.2.4 Parsing with CRF and N-best decoding

Since almost all sequence labelling tasks perform better with conditional random fields (CRF), we decided to give it a go as well along with n-best decoding where the value of $n = 6$. For both CRF and n-best decoding, we use the slightly modified version of *NCRF++* [Yang and Zhang, 2018] – an open-source neural sequence labelling toolkit – that Strzyz et al. [2019b] used. Here instead of the softmax layer which is fed the outputs of the last bi-LSTM, we use CRF as the inference layer. We consider all the features i.e. word embeddings, character embeddings, POS tag embeddings and gaze features for this setup. The baseline system is a model where no gaze features were used. Looking at Table 4.2, we

	Gaze features	dev set UAS	dev set LAS
	baseline	86.41	78.79
L1 norm	2 feats.	86.45	78.73
	17 feats.	**86.60**	**78.89**
	baseline without CRF	86.48	79.03
L1 norm without CRF	2 feats.	86.47	78.68
	17 feats.	**86.91**	**79.20**

Table 4.5: UAS and LAS scores for models trained on all features with CRF and n-best decoding

can see that the *'L1 norm'* provides the best results so we run our experiment only on the gaze features with this norm.

The results for all the experiments are provided in Table 4.5 (best results are in bold). Comparing the values with and without CRF, we can see that the CRF doesn't really improve the scores. There is an overall drop in all the scores including the baseline plus there is a reduction in the improvement with gaze features. Looking at these results, we didn't conduct any further experiments with CRF and n-best decoding or try to improve the CRF code further because ultimately the goal of the thesis was to explore the benefits of gaze features on dependency parsing.

4.3 Experiments with averaged data

In this section, we describe our experiments with data that is averaged over all ten readers of the English part of the Dundee corpus. This section constitutes the core of our findings. For all the experiments in this section, we use a 80-10-10 train-dev-test split with a test set of 241 sentences, dev set of 230 sentences and training set of the remaining 1,897 sentences following Barrett et al. [2016a]. A point to note is that when performing the 80-10-10 data split, we consider the first 10% of the data as our testing data, the next 10% as our dev data and the remaining 80% as the training data so as to avoid any overlap with the testing data and the development data of Section 4.2. Here in the first two sub-sections namely 4.3.1 and 4.3.2 we run the set of experiments on both the sequence labelling parser and the graph-based BIST parser. In all the other sub-sections, the experiments are performed only with the sequence labelling parser.

4.3.1 Parsing with all features

Similar to Section 4.2.1, we first run our sequence labelling parser with all the available features at hand i.e. word features, character features, part-of-speech tags and the gaze features. After that, we run our graph-based parser with the word features, part-of-speech tags and gaze features. For the gaze features, we normalize all the values across the data. For normalizing the values, we use

`scikit-learn`'s normalization from it's pre-processing library with all the three norms i.e. `'l1'`, `'l2'` and `'max'` norm.

We have three variations in terms of the gaze features that we use. In the first case, similar to Section 4.2.1 we use only two normalized gaze features out of the seventeen gaze features listed in Table 4.1 namely, *mean fixation duration* and *total fixation duration*. In the second case we use all seventeen normalized gaze features. Finally we consider a mixture of normalized and raw gaze feature values.

We consider raw values only for those features which deal with the number of fixations or probabilities as mentioned below:

- *n fixations*
- *fixation probability*
- *n re-fixations*
- *re-read probability*
- *w-2 fixation probability*
- *w-1 fixation probability*
- *w+2 fixation probability*
- *w-2 fixation probability*

For the other features listed in Table 4.1, we consider their normalized values. In case of the graph-based parser we use only two variants of the gaze features namely, all seventeen normalized gaze features and mixture of the normalized and raw gaze feature values. We also perform an experiment with the sequence labelling parser where all 17 gaze features are scaled between a value of 0 and 1 using `scikit-learn`'s `MinMaxScaler`. The baseline system in case of both parsers is a model where no gaze features were used.

The results for all the experiments with the sequence labelling parser are provided in Table 4.6 and for the graph-based parser are provided in Table 4.7 (best results are in bold). Similar to the results from Table 4.2, the best results for the sequence labelling parser are obtained by using all 17 features with the L1 norm. There is an improvement of +1.24 for UAS and +0.93 for LAS score as compared to the baseline scores which is more than the improvements we found with a single reader. Curiously enough, for the graph-based parser although the best results are also obtained via the L1 normalization of the gaze features, it is the mixture of raw and normalized gaze features that results in the highest scores. We see an improvement of +0.63 for UAS and +1.11 for LAS score as compared to the baseline scores.

4.3.2 Delexicalized parsing

In this setup, we perform delexicalized parsing of the data. In this case we omit the word and character level features and only use the POS tag embeddings and the gaze features. Similar to Section 4.3.1, we use three different normalization

Gaze features		dev set UAS	dev set LAS
	baseline	83.18	76.45
L1 norm	2 feats.	84.26	77.11
	17 feats.	**84.42**	**77.38**
	Mixture of normalized and raw 17 feats.	83.28	76.78
L2 norm	2 feats.	84.03	**77.38**
	17 feats.	84.30	**77.38**
	Mixture of normalized and raw 17 feats.	83.45	76.82
max norm	2 feats.	83.86	77.09
	17 feats.	83.39	76.66
	Mixture of normalized and raw 17 feats.	83.76	77.15
MinMax Scaler	17 feats.	83.72	76.66

Table 4.6: UAS and LAS scores for models trained on all features for averaged data with sequence labelling parser

Gaze features		dev set UAS	dev set LAS
	baseline	83.41	77.57
L1 norm	17 feats.	83.52	77.68
	Mixture of normalized and raw 17 feats.	**84.04**	**78.68**
L2 norm	17 feats.	82.62	77.24
	Mixture of normalized and raw 17 feats.	83.54	78.16
max norm	17 feats.	83.38	77.66
	Mixture of normalized and raw 17 feats.	83.23	77.85

Table 4.7: UAS and LAS scores for models trained on all features for averaged data with graph-based parser

norms and three sets of gaze features for the sequence labelling parser and two sets of gaze features for the graph-based parser. The baseline system for both the parsers is a model where no gaze features were used and only the POS tag embeddings were fed as input.

Gaze features		dev set	
		UAS	LAS
	baseline	72.83	64.36
L1 norm	2 feats.	73.13	64.22
	17 feats.	73.10	64.88
	Mixture of normalized and raw 17 feats.	73.77	65.55
L2 norm	2 feats.	72.86	64.15
	17 feats.	73.15	64.90
	Mixture of normalized and raw 17 feats.	73.91	**65.73**
max norm	2 feats.	72.71	64.30
	17 feats.	73.94	65.19
	Mixture of normalized and raw 17 feats.	**74.02**	65.71

Table 4.8: UAS and LAS scores for models trained on gaze and POS features for averaged data with sequence labelling parser

Gaze features		dev set	
		UAS	LAS
	baseline	73.83	66.49
L1 norm	17 feats.	73.41	66.90
	Mixture of normalized and raw 17 feats.	**75.10**	**69.28**
L2 norm	17 feats.	73.06	66.81
	Mixture of normalized and raw 17 feats.	74.22	67.99
max norm	17 feats.	74.68	68.34
	Mixture of normalized and raw 17 feats.	74.22	68.50

Table 4.9: UAS and LAS scores for models trained on gaze and POS features for averaged data with graph-based parser

The results for all the experiments with sequence labelling parser are provided in Table 4.8 and for the graph-based parser are provided in Table 4.9 (best results are in bold). Similar to results shown in Table 4.3, the averaged gaze features also seem to help in delexicalized parsing with the sequence labelling parser. The best improvement for the LAS score is +1.37 with the mixture of raw and normalized features with the L2 norm and the best improvement for the UAS score is +1.19 with the mixture of raw and normalized features with max norm. For the graph-based parser the best improvement we get is of +1.27 for the UAS score and +2.79 for the LAS score by using the L1 norm and a mixture of raw and normalized gaze features. Although the highest scores across both parsers vary

amongst all three norms used, the one commonality that can be observed from the results is that using a mixture of raw and normalized gaze features provides the best results when it comes to delexicalized parsing.

4.3.3 Parsing without POS tags

In this setup we perform parsing using the sequence labelling parser without the POS tags by omitting the POS embeddings and only using the word embeddings, character level embeddings and the gaze features. Similar to Section 4.3.1, we use three different normalization norms and three sets of gaze features. The baseline system is a model where no gaze features were used and only the word and character embeddings were fed as input.

Gaze features		dev set	
		UAS	LAS
	baseline	22.22	18.65
L1 norm	2 feats.	22.35	18.71
	17 feats.	22.14	18.48
	Mixture of normalized and raw 17 feats.	21.97	18.19
L2 norm	2 feats.	**22.55**	**18.84**
	17 feats.	22.26	18.40
	Mixture of normalized and raw 17 feats.	22.45	18.65
max norm	2 feats.	22.51	18.75
	17 feats.	22.37	18.73
	Mixture of normalized and raw 17 feats.	22.37	18.55

Table 4.10: UAS and LAS scores for models trained on gaze, word and character features for averaged data

The results for all the experiments are provided in Table 4.10 (best results are in bold). The results in this table do not really tell us much as to how gaze features contribute towards parsing without POS tags. Although there are improvements in some cases for both UAS and LAS scores, these improvements seem to be marginal to make any substantive claim from them.

4.3.4 Parsing as a single task

Up until now, all the experiments that we performed for the sequence labelling parser were with the *combined* MTL setup wherein one of the tasks was to predict the relative position of the head and the other task was to predict the dependency relation. For this set of experiments we try the parser in the ***single*** MTL setup where we have a single task in which we predict both the relative position as well as the dependency relation. We consider all the features – word embeddings, character embeddings, POS tag embeddings and gaze features – for these experiments while using the normalized values for all seventeen gaze features. The baseline model doesn't use any of the gaze features.

Gaze features	dev set	
	UAS	LAS
baseline + single	82.49	76.18
17 feats. + L1 norm + single	**82.82**	**76.82**
17 feats. + L2 norm + single	82.55	76.30
17 feats. + max norm + single	82.06	75.33
baseline + combined	83.18	76.45
17 feats. + L1 norm + combined	**84.42**	**77.38**
17 feats. + L2 norm + combined	84.30	**77.38**
17 feats. + max norm + combined	83.39	76.66

Table 4.11: UAS and LAS scores for models trained on all features for averaged data in ***single*** MTL setup

The results for all the experiments are provided in Table 4.11 (best results are in bold). On comparing with results of the combined MTL setup, we can see that using the model in a single MTL setup degrades the results in general and this corroborates the findings of Strzyz et al. [2019a] where they say that the combined MTL setup offers the best results.

4.3.5 Assessing the impact of various gaze features on sequence labelling parsing with all features

Looking at Table 4.6, we can see that the model where the data is normalized by the *L1 norm* with all 17 gaze features performs the best when all the features are in use. To get some sort of an idea as to how much individual gaze features and grouped gaze features as per their category (i.e. basic, early, late and context) contribute towards parsing, we compare scores from the baseline model where no gaze features are used to models with individual gaze features or grouped gaze features.

Table 4.12 shows the results with best scores marked in bold. The results from the dev set show that the grouped *context* gaze features provide the best improvements as compared to the baseline with an improvement of +0.99 for UAS score and +0.73 for the LAS score. Individually, *first fixation duration* provides almost the same improvements as the grouped *context* features with an improvement of +0.97 for UAS score and +0.73 for LAS score. These results are in line with the results of Strzyz et al. [2019b] which are shown in Table 4.13, who found improvements with the *early* and *context* grouped gaze features. On comparing the two tables 4.12 and 4.13 we see that Strzyz et al. [2019b] achieve higher baseline results and our only estimate for this is that they managed to get the CRF layer working properly for them boosting their result. We do however note that our improvements with gaze features are higher than that of Strzyz et al. [2019b] and this could be because we directly incorporate the gaze features instead of predicting them as an auxiliary task.

	Gaze features	dev set UAS	dev set LAS
	baseline	83.18	76.45
Basic	n fixations	+0.60	+0.31
	fixation probability	+0.87	+0.46
	mean fixation duration	+0.62	+0.19
	total fixation duration	+0.68	+0.08
	basic features	+0.91	+0.50
Early	first fixation duration	+0.97	**+0.73**
	first pass duration	+0.83	+0.56
	early features	+0.77	+0.62
Late	n re-fixations	+0.60	+0.19
	re-read probability	+0.25	+0.35
	late features	+0.85	+0.70
Context	total regression from duration	+0.27	+0.23
	w-2 fixation probability	+0.50	+0.33
	w-1 fixation probability	+0.60	+0.08
	w+1 fixation probability	+0.48	-0.06
	w+2 fixation probability	+0.50	+0.33
	w-2 fixation duration	+0.87	+0.14
	w-1 fixation duration	+0.39	+0.23
	w+1 fixation duration	+0.77	+0.50
	w+2 fixation duration	+0.18	+0.14
	context features	**+0.99**	**+0.73**

Table 4.12: Impact of various gaze features on sequence labelling dependency parsing with all features. The values reflect improvement or deterioration over the corresponding baseline scores.

	Gaze features	dev set UAS	dev set LAS
	baseline	85.36	79.40
Basic	*n* fixations	-0.04	-0.11
	fixation probability	-0.04	+0.17
	mean fixation duration	-0.15	-0.02
	total fixation duration	-0.02	-0.05
	basic features	0.00	+0.17
Early	first fixation duration	-0.06	+0.06
	first pass duration	+0.14	+0.09
	early features	**+0.25**	+0.17
Late	*n* re-fixations	+0.16	-0.15
	re-read probability	-0.02	+0.17
	late features	+0.18	+0.24
Context	w-1 fixation probability	-0.19	+0.07
	w+1 fixation probability	0.00	-0.33
	w-1 fixation duration	+0.07	+0.28
	w+1 fixation duration	+0.03	+0.13
	context features	**+0.25**	**+0.32**

Table 4.13: Impact of various gaze features when used as auxiliary task for sequence labelling parsing. The values reflect improvement or deterioration over the corresponding baseline scores and have been taken from Strzyz et al. [2019b]

4.3.6 Assessing the impact of various gaze features on delexicalized sequence labelling parsing

Looking at Table 4.8, we can see that the model where the data is a mixture of raw and normalized values with the *max norm* is more or less the best in terms of performance when it comes to delexicalized parsing. To get some sort of an idea as to how much individual gaze features and grouped gaze features as per their category (i.e. basic, early, late and context) contribute towards delexicalized parsing, we compare scores from the baseline model where no gaze features are used to models with individual gaze features or grouped gaze features.

A point to note is that, whenever we make use of the following features, we are considering their raw values and not normalized values.

- *n fixations*
- *fixation probability*
- *n re-fixations*
- *re-read probability*
- *w-2 fixation probability*
- *w-1 fixation probability*
- *w+2 fixation probability*
- *w-2 fixation probability*

The rest of the gaze features when used are normalized with the *max norm*.

Table 4.14 shows the results with best scores marked in bold. Interestingly, the results from the dev set show that in case of delexicalized parsing, the set of *basic* gaze features, grouped together provide the best improvements as compared to the baseline. They provide an improvement of +0.71 for the UAS score and +0.42 for the LAS score. Individually, the *w+2 fixation duration* and *n fixations* features seem to provide the best improvements over the baseline.

	Gaze features	dev set UAS	dev set LAS
	baseline	72.83	64.36
Basic	*n* fixations	+0.34	+0.23
	fixation probability	+0.09	-0.02
	mean fixation duration	-0.04	-0.04
	total fixation duration	-0.06	-0.14
	basic features	**+0.71**	**+0.42**
Early	first fixation duration	-0.12	-0.37
	first pass duration	-0.02	+0.04
	early features	+0.25	0.00
Late	*n* re-fixations	+0.32	+0.15
	re-read probability	+0.07	+0.08
	late features	+0.11	-0.08
Context	total regression from duration	+0.19	-0.14
	w-2 fixation probability	-0.12	-0.39
	w-1 fixation probability	+0.13	-0.52
	w+1 fixation probability	-0.16	-0.10
	w+2 fixation probability	-0.12	-0.39
	w-2 fixation duration	+0.30	-0.10
	w-1 fixation duration	-0.18	-0.29
	w+1 fixation duration	+0.03	-0.29
	w+2 fixation duration	+0.34	+0.25
	context features	+0.48	+0.04

Table 4.14: Impact of various gaze features on delexicalized parsing. The values reflect improvement or deterioration over the corresponding baseline scores.

5. Evaluation

In this chapter, we evaluate the best setups from Section 4.3 on the test data and compare it with the results of the baseline setup.

5.1 Evaluating parsing with all features for averaged data on test set

	Gaze features	test set UAS	test set LAS
	baseline	82.19	75.82
L1 norm	17 feats.	**82.46**	**75.93**

Table 5.1: Evaluation of sequence labelling parser with all features on test set with UAS and LAS scores (best results in bold).

	Gaze features	test set UAS	test set LAS
	baseline	**82.29**	76.16
L1 norm	Mixture of normalized and raw 17 feats.	81.90	**76.31**

Table 5.2: Evaluation of graph-based parser with all features on test set with UAS and LAS scores (best results in bold).

From Table 4.6 and Table 4.7, it is clear that the setup using all 17 gaze features with L1 norm is the best setup for the sequence labelling parser and using a mixture of normalized and raw gaze features with L1 norm is the best setup for the graph-based parser. Thus, we run these setups on the test data and compare the results with the baseline parser and each other.

Table 5.1 shows that for the sequence labelling parser although there is an improvement of +0.27 for the UAS score and +0.11 for the LAS score, these improvements are quite small and not statistically significant. From Table 5.2 we see that for the graph-based parser, the performance actually deteriorates by -0.39 for the UAS score as compared to the baseline and for the LAS score we see an improvement of +0.15 as compared to the baseline although this improvement is also statistically insignificant. On comparison with their corresponding scores on the development set, these improvements aren't as large and the performance even deteriorates for the graph-based parser in terms of the UAS score leading us to believe that although the gaze features do help marginally in parsing at least for improving the LAS score, the improvement that we saw with the development data was more due to the data and not because of the gaze features.

Barrett et al. [2015] in their work on creating the Dundee Treebank ran a graph-based parser called *mate-tools* [Bohnet, 2010] with its default settings. The

parser got a LAS score of 82.23 and UAS score of 85.06 wherein they used a 5-fold 80-20 cross validation since there is no separate test data available for the Dundee Treebank. Meanwhile, the baseline parser by Strzyz et al. [2019b] obtained a UAS score of 84.37 and LAS score of 78.24 on the test set. Over all, our baseline parser scores are lower than the other parsers on Dundee Treebank.

5.2 Evaluating delexicalized parser for averaged data on test set

	Gaze features	test set UAS	test set LAS
	baseline	72.86	64.89
max norm	Mixture of normalized and raw 17 feats.	**74.11****	**65.84***

Table 5.3: Evaluation of delexicalized sequence labelling parser on test set with UAS and LAS scores (best results in bold; *, ** indicates significant improvement over the baseline; * $p < 0.05$, ** $p < 0.01$ McNemar's test).

	Gaze features	test set UAS	test set LAS
	baseline	74.14	66.80
L1 norm	Mixture of normalized and raw 17 feats.	**74.89**	**68.69****

Table 5.4: Evaluation of delexicalized graph-based parser on test set with UAS and LAS scores (best results in bold; ** indicates significant improvement over the baseline; ** $p < 0.01$ McNemar's test).

Table 4.8 shows us that the max norm with mixture of normalized and raw gaze features gives us the highest UAS score and the L2 norm with mixture of normalized and raw gaze features gives the highest LAS score on the development set for the sequence labelling parser. However since the L2 norm gives a LAS score of 65.73 and the max norm gives a very close LAS score of 65.71, we shall just consider the max norm as the best setup and compare its results on the test data with the baseline. From Table 4.9 it is clear that the setup using mixture of normalized and raw gaze features with L1 norm is the best setup for the graph-based parser.

The results in Table 5.3 and Table 5.4 seem to indicate that eye-tracking features most definitely help in delexicalized parsing. There is an improvement of +1.25 over the baseline for the UAS score and an improvement of +0.95 over the baseline for the LAS score, both of which are statistically significant for the sequence labelling parser. As for the graph-based parser, there is an improvement of +0.75 for the UAS score and +1.89 for the LAS score over the baseline with the improvement for LAS score being statistically significant. The sequence labelling

parser seems to give better scores for UAS whereas the graph-based parser seems to improve the LAS scores more.

Deprel	Counts
nmod	635
case	613
det	576
amod	353
nsubj	307
advmod	248
mark	241
dobj	184
root	175
cc	173

Table 5.5: Most common correctly predicted deprels by our delexicalized sequence labelling parser

Deprel	Counts	Relative frequency
root	23	9.50%
nmod	23	3.20%
aux	21	14.68%
nsubj	15	3.93%
conj	13	6.22%
case	13	1.96%
cop	12	12.76%
det	11	1.86%
dobj	10	4.11%
mark	9	3.10%

Table 5.6: Most common correctly predicted deprels by our delexicalized sequence labelling parser compared to baseline

Table 5.5 and Table 5.6 show the 10 most common correctly predicted dependency relations by our gaze augmented sequence labelling dependency parser. Table 5.6 in particular shows those dependency relations correctly identified by our parser which the baseline parser couldn't. The dependency relations towards whose identification the gaze features help the most are *root*, *nmod* and *aux*.

Table 5.7 and Table 5.8 show us that gaze features help in correctly identifying the head and dependency relation the most for *NOUNS* and *VERBS*. The high relative improvements for *root*, *aux*, *cop* and *VERB* makes us conclude that gaze features contain some important syntactic information related to verbs allowing the parser to parse the verbs much better. As verbs are heads of syntactic clauses, it may also be that gaze features help the parser distinguish the main clause (headed by a verb with the 'root' label) from subordinate clauses. The relative improvement for *CONJ* is also high meaning gaze features somehow help the parser in understanding co-ordination structures.

POS tag	Counts	Relative frequency
NOUN	113	7.95%
VERB	75	7.87%
CONJ	25	13.81%
ADP	18	2.46%
DET	17	2.51%
ADV	16	4.24%
PRT	15	6.85%
ADJ	15	3.31%
PRON	14	4.44%
NUM	4	5.71%

Table 5.7: Most common POS tags with correctly predicted heads by our delexicalized sequence labelling parser compared to baseline

POS tag	Counts	Relative frequency
NOUN	112	7.88%
VERB	82	8.61%
CONJ	27	14.91%
ADV	21	5.57%
ADP	19	2.60%
DET	17	2.51%
PRON	16	5.08%
ADJ	15	3.31%
PRT	14	6.40%
NUM	3	4.28%

Table 5.8: Most common POS tags with correctly predicted heads and deprels by our delexicalized sequence labelling parser compared to baseline

Conclusion

In this thesis, we explored the benefits of using eye-tracking features for dependency parsing. We cast dependency parsing as a sequence labelling problem for the English part of the Dundee Corpus and we also augmented a graph-based parser with gaze features. We performed a set of experiments wherein we tried different parser settings and different eye-tracking feature selection along with various feature normalization techniques to try to answer the questions posed by us at the beginning of the thesis.

Our experiments show that although eye-tracking features do help in dependency parsing where all features (word level, character level and POS tags) are used for the LAS score, the improvements over the baseline are not statistically significant. Using the L1 norm for normalizing the eye-tracking features seems to provide the best results in this setup. This answers the first question posed by us which was whether eye-tracking features help in dependency parsing and to what extent.

Answering our second research question which was to determine the benefits of gaze features in delexicalized parsing if any, our experimental results show that there is statistically significant improvement over the baseline when we use eye-tracking features for delexicalized parsing. The improvements are significant for both the UAS and the LAS scores in case of the sequence labelling parser and significant only for the LAS score in case of the graph-based parser. For this particular setup, our results seem to indicate that using a mixture of raw and normalized eye-tracking features seems to provide the best improvements.

When it comes to incorporating gaze features with a parser using only lexical features (word level and character level), our experiments indicate an improvement in some cases for both the UAS and LAS scores. However these improvements are only marginal and not consistent and hence, we cannot make any substantive claim to answer the third question posed by us. Another takeaway from our experiments is that a combined multi-tasking approach performs better than predicting the result as a single label. The single setup degrades the results in general as compared to the combined multi-task setup even including the baseline results.

The final question we answer is about the contribution of specific gaze features towards dependency parsing. When it comes to parsing with all features, the grouped *context* gaze features provide the best improvements whereas individually the *first fixation duration* provides the maximum improvement. On the other hand, for delexicalized parsing, the grouped *basic* gaze features provide the best improvements and individually, the *w+2 fixation duration* and *n fixations* features seem to improve the parser the most. The results also seem to indicate that grouped gaze features perform better than individual gaze features and that only on combining different gaze feature groups can we obtain significant improvements over the baseline if any. This seems to suggest that the various eye-tracking features contain information which is complementary in nature.

Future work

While we worked with token-level gaze features in our thesis, an important future work could be to leverage type-aggregated gaze features for dependency parsing. With this we can leverage eye-tracking data at just the training time and not require recording eye-tracking data at test time. Also, other works have indicated that type-aggregated features seem to perform better than token-level features with the idea that eye-movement data is quite noisy and averaging over all tokens of a type reduces the noise more than just averaging over all participants that read each token. An additional potential benefit of leveraging type-aggregated gaze features is also that it could help a dependency parser in cross-domain settings.

Another interesting avenue to explore would be to determine if gaze and syntax co-relations can be transferred across a pair of languages, effectively helping us in creating a cross-lingual dependency parser. Since one of our key findings in this thesis was that eye-tracking features significantly improve delexicalized parsing and delexicalized parsing is useful in low-resource and cross-lingual settings, this is definitely something we plan on exploring in the future.

Bibliography

Sally Andrews, Brett Miller, and Keith Rayner. Eye movements and morphological segmentation of compound words: There is a mouse in mousetrap. *European Journal of Cognitive Psychology - EUR J COGN PSYCHOL*, 16:285–311, 01 2004. doi: 10.1080/09541440340000123.

Lauriane Aufrant, Guillaume Wisniewski, and François Yvon. Zero-resource dependency parsing: Boosting delexicalized cross-lingual transfer with linguistic knowledge. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 119–130, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee. URL `https://www.aclweb.org/anthology/C16-1012`.

Michael Auli and Jianfeng Gao. Decoder integration and expected BLEU training for recurrent neural network language models. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 136–142, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-2023. URL `https://www.aclweb.org/anthology/P14-2023`.

Miguel Ballesteros, Chris Dyer, and Noah A. Smith. Improved transition-based parsing by modeling characters instead of words with LSTMs. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 349–359, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1041. URL `https://www.aclweb.org/anthology/D15-1041`.

Maria Barrett and Anders Søgaard. Reading behavior predicts syntactic categories. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 345–349, Beijing, China, July 2015a. Association for Computational Linguistics. doi: 10.18653/v1/K15-1038. URL `https://www.aclweb.org/anthology/K15-1038`.

Maria Barrett and Anders Søgaard. Using reading behavior to predict grammatical functions. In *Proceedings of the Sixth Workshop on Cognitive Aspects of Computational Language Learning*, pages 1–5, Lisbon, Portugal, September 2015b. Association for Computational Linguistics. doi: 10.18653/v1/W15-2401. URL `https://www.aclweb.org/anthology/W15-2401`.

Maria Barrett, Joachim Bingel, Frank Keller, and Anders Søgaard. Weakly supervised part-of-speech tagging using eye-tracking data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 579–584, Berlin, Germany, August 2016a. Association for Computational Linguistics. doi: 10.18653/v1/P16-2094. URL `https://www.aclweb.org/anthology/P16-2094`.

Maria Barrett, Frank Keller, and Anders Søgaard. Cross-lingual transfer of correlations between parts of speech and gaze features. In *Proceedings of COLING*

2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 1330–1339, Osaka, Japan, December 2016b. The COLING 2016 Organizing Committee. URL https://www.aclweb.org/anthology/C16-1126.

Maria Barrett, Joachim Bingel, Nora Hollenstein, Marek Rei, and Anders Søgaard. Sequence classification with human attention. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 302–312, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/K18-1030. URL https://www.aclweb.org/anthology/K18-1030.

Maria Jung Barrett, Zeljko Agic, and Anders Søgaard. The dundee treebank. In *Proceedings of the Fourteenth International Workshop on Treebanks and Linguistic Theories*, pages 242–248. Association for Computational Linguistics, 2015. ISBN 978-83-63159-18-4.

Joachim Bingel, Maria Barrett, and Sigrid Klerke. Predicting misreadings from gaze in children with reading difficulties. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 24–34, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-0503. URL https://www.aclweb.org/anthology/W18-0503.

Bernd Bohnet. Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 89–97, Beijing, China, August 2010. Coling 2010 Organizing Committee. URL https://www.aclweb.org/anthology/C10-1011.

Barbara A. Brooks, Dianne M. K. Impelman, and Janet T. Lum. Backward and forward masking associated with saccadic eye movement. *Perception & Psychophysics*, 30(1):62–70, Jan 1981. ISSN 1532-5962. doi: 10.3758/BF03206137. URL https://doi.org/10.3758/BF03206137.

Marc Brysbaert and Françoise Vitu. *Word skipping: Implications for theories of eye movement control in reading.*, pages 125–147. Eye guidance in reading and scene perception. Elsevier Science Ltd, Oxford, England, 1998. ISBN 0-08-043361-8 (Hardcover). doi: 10.1016/B978-008043361-5/50007-9. URL https://doi.org/10.1016/B978-008043361-5/50007-9.

Patricia A. Carpenter and Marcel Adam Just. What your eyes do while your mind is reading. In KEITH RAYNER, editor, *Eye Movements in Reading*, pages 275 – 307. Academic Press, 1983. ISBN 978-0-12-583680-7. doi: https://doi.org/10.1016/B978-0-12-583680-7.50022-9. URL http://www.sciencedirect.com/science/article/pii/B9780125836807500229.

Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, Jul 1997. ISSN 1573-0565. doi: 10.1023/A:1007379606734. URL https://doi.org/10.1023/A:1007379606734.

Roger Chaffin, Robin Morris, and Rachel Seely. Learning new word meanings from context: A study of eye movements. *Journal of experimental psychology. Learning, memory, and cognition*, 27:225–35, 02 2001. doi: 10.1037/0278-7393.27.1.225.

Eugene Chekaluk and Keith R. Llewellyn. Visual stimulus input, saccadic suppression, and detection of information from the postsaccade scene. *Perception & Psychophysics*, 48(2):135–142, Mar 1990. ISSN 1532-5962. doi: 10.3758/BF03207080. URL https://doi.org/10.3758/BF03207080.

Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1082. URL https://www.aclweb.org/anthology/D14-1082.

Joe Cheri, Abhijit Mishra, and Pushpak Bhattacharyya. Leveraging annotators' gaze behaviour for coreference resolution. In *Proceedings of the 7th Workshop on Cognitive Aspects of Computational Language Learning*, pages 22–26, Berlin, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-1904. URL https://www.aclweb.org/anthology/W16-1904.

N. Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 2(3):113–124, 1956.

Uschi Cop, Nicolas Dirix, Denis Drieghe, and Wouter Duyck. Presenting geco: An eyetracking corpus of monolingual and bilingual sentence reading. *Behavior Research Methods*, 49(2):602–615, 2017. ISSN 1554-3528. doi: 10.3758/s13428-016-0734-0. URL https://doi.org/10.3758/s13428-016-0734-0.

Michael A. Covington. A fundamental algorithm for dependency parsing. In *In Proceedings of the 39th Annual ACM Southeast Conference*, pages 95–102, 2001.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Timothy Dozat and Christopher D. Manning. Deep biaffine attention for neural dependency parsing, 2016.

Susan A Duffy, Robin K Morris, and Keith Rayner. Lexical ambiguity and fixation times in reading. *Journal of Memory and Language*, 27(4):429 – 446, 1988. ISSN 0749-596X. doi: https://doi.org/10.1016/0749-596X(88)90066-6. URL http://www.sciencedirect.com/science/article/pii/0749596X88900666.

Greg Durrett and Dan Klein. Neural CRF parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 302–312, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1030. URL https://www.aclweb.org/anthology/P15-1030.

Jay Earley. An efficient context-free parsing algorithm. *Commun. ACM*, 13(2): 94–102, February 1970. ISSN 0001-0782. doi: 10.1145/362007.362035. URL https://doi.org/10.1145/362007.362035.

Susan F. Ehrlich and Keith Rayner. Contextual effects on word perception and eye movements during reading. *Journal of Verbal Learning and Verbal Behavior*, 20(6):641 – 655, 1981. ISSN 0022-5371. doi: https://doi.org/10.1016/S0022-5371(81)90220-6. URL http://www.sciencedirect.com/science/article/pii/S0022537181902206.

Jeffrey L. Elman. Finding structure in time. *Cogn. Sci.*, 14:179–211, 1990.

Daniel Fernández-González and Carlos Gómez-Rodríguez. Left-to-right dependency parsing with pointer networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 710–716, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1076. URL https://www.aclweb.org/anthology/N19-1076.

Lyn Frazier and Keith Rayner. Making and correcting errors during sentence comprehension: Eye movements in the analysis of structurally ambiguous sentences. *Cognitive Psychology*, 14(2):178 – 210, 1982. ISSN 0010-0285. doi: https://doi.org/10.1016/0010-0285(82)90008-1. URL http://www.sciencedirect.com/science/article/pii/0010028582900081.

Lyn Frazier and Keith Rayner. Resolution of syntactic category ambiguities: Eye movements in parsing lexically ambiguous sentences. *Journal of Memory and Language*, 26(5):505 – 526, 1987. ISSN 0749-596X. doi: https://doi.org/10.1016/0749-596X(87)90137-9. URL http://www.sciencedirect.com/science/article/pii/0749596X87901379.

M. Gareth Gaskell, Adrian Staub, and Keith Rayner. Eye movements and on-line comprehension processes, 09 2012. URL https://www.oxfordhandbooks.com/view/10.1093/oxfordhb/9780198568971.001.0001/oxfordhb-9780198568971-e-019.

Vincent Gautier, J. O'Regan, and Jean Gargasson. 'the-skipping' revisited in french: Programming saccades to skip the article 'les'. *Vision research*, 40: 2517–31, 02 2000. doi: 10.1016/S0042-6989(00)00089-4.

Carlos Gómez-Rodríguez and David Vilares. Constituent parsing as sequence labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1314–1324, Brussels, Belgium, October-November

2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1162. URL https://www.aclweb.org/anthology/D18-1162.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Nora Hollenstein and Ce Zhang. Entity recognition at first sight: Improving NER with eye movement information. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1–10, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1001. URL https://www.aclweb.org/anthology/N19-1001.

Nora Hollenstein, Jonathan Rotsztejn, Marius Troendle, Andreas Pedroni, Ce Zhang, and Nicolas Langer. Zuco, a simultaneous eeg and eye-tracking resource for natural sentence reading. *Scientific Data*, 5(1):180291, 2018. ISSN 2052-4463. doi: 10.1038/sdata.2018.291. URL https://doi.org/10.1038/sdata.2018.291.

Nora Hollenstein, Antonio de la Torre, Nicolas Langer, and Ce Zhang. CogniVal: A framework for cognitive word embedding evaluation. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 538–549, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/K19-1050. URL https://www.aclweb.org/anthology/K19-1050.

Jukka Hyönä and Alexander Pollatsek. Reading finnish compound words: Eye fixations are affected by component morphemes. *Journal of Experimental Psychology: Human Perception and Performance*, 24(6):1612–1627, 1998. doi: 10.1037/0096-1523.24.6.1612. URL https://doi.org/10.1037/0096-1523.24.6.1612.

Jukka Hyönä, Raymond Bertram, and Alexander Pollatsek. Are long compound words identified serially via their constituents? evidence from an eye-movement-contingent display change study. *Memory & cognition*, 32(4): 523–532, June 2004. ISSN 0090-502X. doi: 10.3758/bf03195844. URL https://doi.org/10.3758/bf03195844.

Albrecht Werner Inhoff and Keith Rayner. Parafoveal word processing during eye fixations in reading: Effects of word frequency. *Perception & Psychophysics*, 40(6):431–439, Nov 1986. ISSN 1532-5962. doi: 10.3758/BF03208203. URL https://doi.org/10.3758/BF03208203.

Tao Ji, Yuanbin Wu, and Man Lan. Graph-based dependency parsing with graph neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2475–2485, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1237. URL https://www.aclweb.org/anthology/P19-1237.

Drahomíra johanka Spoustová and Miroslav Spousta. Dependency parsing as a sequence labeling task. In *Prague Bull. Math. Linguistics*, 2010.

Barbara Juhasz and Keith Rayner. The role of age of acquisition and word frequency in reading: Evidence from eye fixation durations. *Visual Cognition - VIS COGN*, 13:846–863, 05 2006. doi: 10.1080/13506280544000075.

Barbara Juhasz, Matthew Starr, Albrecht Inhoff, and Lars Placke. The effects of morphology on the processing of compound words: Evidence from naming, lexical decisions and eye fixations. *British journal of psychology (London, England : 1953)*, 94:223–44, 06 2003. doi: 10.1348/000712603321661903.

Barbara J. Juhasz. Age-of-acquisition effects in word and picture identification. *Psychological Bulletin*, 131(5):684–712, 2005. doi: 10.1037/0033-2909.131.5.684. URL `https://doi.org/10.1037/0033-2909.131.5.684`.

Barbara J. Juhasz and Keith Rayner. Investigating the effects of a set of intercorrelated variables on eye fixation durations in reading. *Journal of experimental psychology. Learning, memory, and cognition*, 29 6:1312–8, 2003.

Marcel Adam Just and Patricia Carpenter. A theory of reading: from eye fixations to comprehension. *Psychological review*, 87 4:329–54, 1980.

R. M. Kaplan. 'a general syntactic processor'. In *R. Rustin, Natural Language Processing,*, pages 193–241. Algorithmics Press, 1973.

T. Kasami. An efficient recognition and syntax analysis algorithm for context-free languages. Technical Report AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, MA†, 1965.

M Kay. *Algorithm Schemata and Data Structures in Syntactic Processing*, page 35–70. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1986. ISBN 0934613117.

Alan Kennedy, Robin Hill, and Jöel Pynte. The dundee corpus. In *Proceedings of the European Conference on Eye Movements (ECEM)*, 2003.

Eliyahu Kiperwasser and Yoav Goldberg. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327, 2016. doi: 10.1162/tacl_a_00101. URL `https://www.aclweb.org/anthology/Q16-1023`.

Nikita Kitaev and Dan Klein. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1249. URL `https://www.aclweb.org/anthology/P18-1249`.

Sigrid Klerke, Sheila Castilho, Maria Barrett, and Anders Søgaard. Reading metrics for estimating task efficiency with MT output. In *Proceedings of the Sixth Workshop on Cognitive Aspects of Computational Language Learning*, pages 6–13, Lisbon, Portugal, September 2015. Association for Computational

Linguistics. doi: 10.18653/v1/W15-2402. URL https://www.aclweb.org/anthology/W15-2402.

Sigrid Klerke, Yoav Goldberg, and Anders Søgaard. Improving sentence compression by learning to predict gaze. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1528–1533, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1179. URL https://www.aclweb.org/anthology/N16-1179.

K. Krafka, A. Khosla, P. Kellnhofer, H. Kannan, S. Bhandarkar, W. Matusik, and A. Torralba. Eye tracking for everyone. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2176–2184, 2016.

Zuchao Li, Jiaxun Cai, Shexia He, and Hai Zhao. Seq2seq dependency parsing. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3203–3214, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1271.

Simon P. Liversedge, Keith Rayner, Sarah J. White, Dorine Vergilino-Perez, John M. Findlay, and Robert W. Kentridge. Eye movements when reading disappearing text: is there a gap effect in reading? *Vision Research*, 44(10): 1013 – 1024, 2004. ISSN 0042-6989. doi: https://doi.org/10.1016/j.visres.2003.12.002. URL http://www.sciencedirect.com/science/article/pii/S0042698903007909.

Alessandro Lopopolo, Stefan L. Frank, Antal van den Bosch, and Roel Willems. Dependency parsing with your eyes: Dependency structure predicts eye regressions during reading. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, pages 77–85, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-2909. URL https://www.aclweb.org/anthology/W19-2909.

Kristian Lukander, Sharman Jagadeesan, Huageng Chi, and Kiti Müller. Omg! a new robust, wearable and affordable open source mobile gaze tracker. In *Proceedings of the 15th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '13, page 408–411, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450322737. doi: 10.1145/2493190.2493214. URL https://doi.org/10.1145/2493190.2493214.

Steven G. Luke and Kiel Christianson. The provo corpus: A large eye-tracking corpus with predictability norms. *Behavior Research Methods*, 50(2):826–833, 2018. ISSN 1554-3528. doi: 10.3758/s13428-017-0908-4. URL https://doi.org/10.3758/s13428-017-0908-4.

Sandeep Mathias, Diptesh Kanojia, Kevin Patel, Samarth Agrawal, Abhijit Mishra, and Pushpak Bhattacharyya. Eyes are the windows to the soul: Predicting the rating of text quality using gaze behaviour. In *Proceedings of the*

56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2352–2362, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1219. URL https://www.aclweb.org/anthology/P18-1219.

George W. McConkie and Keith Rayner. The span of the effective stimulus during a fixation in reading. *Perception & Psychophysics*, 17(6):578–586, Nov 1975. ISSN 1532-5962. doi: 10.3758/BF03203972. URL https://doi.org/10.3758/BF03203972.

Ryan Mcdonald. *Discriminative Learning and Spanning Tree Algorithms for Dependency Parsing*. PhD thesis, USA, 2006. AAI3225503.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada, October 2005. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/H05-1066.

Enrique Meseguer, Manuel Carreiras, and Charles Clifton. Overt reanalysis strategies and eye movements during the reading of mild garden path sentences. *Memory & cognition*, 30:551–61, 07 2002. doi: 10.3758/BF03194956.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013b.

Abhijit Mishra, Diptesh Kanojia, and Pushpak Bhattacharyya. Predicting readers' sarcasm understandability by modeling gaze behavior. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 3747–3753. AAAI Press, 2016a.

Abhijit Mishra, Diptesh Kanojia, Seema Nagar, Kuntal Dey, and Pushpak Bhattacharyya. Harnessing cognitive features for sarcasm detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1095–1104, Berlin, Germany, August 2016b. Association for Computational Linguistics. doi: 10.18653/v1/P16-1104. URL https://www.aclweb.org/anthology/P16-1104.

Abhijit Mishra, Diptesh Kanojia, Seema Nagar, Kuntal Dey, and Pushpak Bhattacharyya. Leveraging cognitive features for sentiment analysis. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 156–166, Berlin, Germany, August 2016c. Association for Computational Linguistics. doi: 10.18653/v1/K16-1016. URL https://www.aclweb.org/anthology/K16-1016.

Abhijit Mishra, Diptesh Kanojia, Seema Nagar, Kuntal Dey, and Pushpak Bhattacharyya. Scanpath complexity: Modeling reading effort using gaze information. In *AAAI Conference on Artificial Intelligence*, 2017. URL https://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14867.

Robin K. Morris. Lexical and message-level sentence context effects on fixation times in reading. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 20(1):92–103, 1994. doi: 10.1037/0278-7393.20.1.92. URL https://doi.org/10.1037/0278-7393.20.1.92.

Joakim Nivre. An efficient algorithm for projective dependency parsing. In *Proceedings of the Eighth International Conference on Parsing Technologies*, pages 149–160, Nancy, France, April 2003. URL https://www.aclweb.org/anthology/W03-3017.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL https://www.aclweb.org/anthology/L16-1262.

Wenzhe Pei, Tao Ge, and Baobao Chang. An effective neural network model for graph-based dependency parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 313–322, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1031. URL https://www.aclweb.org/anthology/P15-1031.

Alexander Pollatsek and Jukka Hyönä. The role of semantic transparency in the processing of finnish compound words. *Language and Cognitive Processes*, 20 (1):261–290, 2005. doi: 10.1080/01690960444000098. URL https://doi.org/10.1080/01690960444000098.

Alexander Pollatsek, Gary E. Raney, Linda Lagasse, and Keith Rayner. The use of information below fixation in reading and in visual search. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, 47 (2):179–200, 1993. doi: 10.1037/h0078824. URL https://doi.org/10.1037/h0078824.

Alexander Pollatsek, Jukka Hyönä, and Raymond Bertram. The role of morphological constituents in reading finnish compound words. *Journal of experimental psychology. Human perception and performance*, 26:820–33, 05 2000. doi: 10.1037/0096-1523.26.2.820.

K Rayner and JH Bertera. Reading without a fovea. *Science*, 206(4417):468–469, 1979. ISSN 0036-8075. doi: 10.1126/science.504987. URL https://science.sciencemag.org/content/206/4417/468.

Keith Rayner. Eye movements in reading and information processing: 20 years of research. *Psychological bulletin*, 124 3:372–422, 1998.

Keith Rayner and Susan A. Duffy. Lexical complexity and fixation times in reading: Effects of word frequency, verb complexity, and lexical ambiguity. *Memory & Cognition*, 14(3):191–201, 1986. ISSN 1532-5946. doi: 10.3758/BF03197692. URL https://doi.org/10.3758/BF03197692.

Keith Rayner and Louise Frazier. Selection mechanisms in reading lexically ambiguous words. *Journal of experimental psychology. Learning, memory, and cognition*, 15 5:779–90, 1989.

Keith Rayner and George W. McConkie. What guides a reader's eye movements? *Vision Research*, 16(8):829–837, 1976. ISSN 0042-6989(Print). doi: 10.1016/0042-6989(76)90143-7. URL https://doi.org/10.1016/0042-6989(76)90143-7.

Keith Rayner and Arnold D. Well. Effects of contextual constraint on eye movements in reading: A further examination. *Psychonomic Bulletin & Review*, 3(4):504–509, Dec 1996. ISSN 1531-5320. doi: 10.3758/BF03214555. URL https://doi.org/10.3758/BF03214555.

Keith Rayner, Albrecht Werner Inhoff, Robert E. Morrison, Maria Louisa Slowiaczek, and James H. Bertera. Masking of foveal and parafoveal vision during eye fixations in reading. *Journal of experimental psychology. Human perception and performance*, 7 1:167–79, 1981.

Keith Rayner, Sara Sereno, and Gary Raney. Eye movement control in reading: A comparison of two types of models. *Journal of experimental psychology. Human perception and performance*, 22:1188–200, 11 1996. doi: 10.1037/0096-1523.22.5.1188.

Keith Rayner, Gretchen Kambe, and Susan A. Duffy. The effect of clause wrap-up on eye movements during reading. *The Quarterly Journal of Experimental Psychology Section A*, 53(4):1061–1080, 2000. doi: 10.1080/713755934. URL https://doi.org/10.1080/713755934.

Keith Rayner, Simon P. Liversedge, Sarah Jane White, and Dorine Vergilino-Perez. Reading disappearing text. *Psychological Science*, 14:385 – 388, 2003.

Keith Rayner, Simon P. Liversedge, and Sarah J. White. Eye movements when reading disappearing text: The importance of the word to the right of fixation. *Vision Research*, 46(3):310 – 323, 2006. ISSN 0042-6989. doi: https://doi.org/10.1016/j.visres.2005.06.018. URL http://www.sciencedirect.com/science/article/pii/S0042698905003068.

Sebastian Ruder. An overview of multi-task learning in deep neural networks, 2017.

Javier San Agustin, Henrik Skovsgaard, John Paulin Hansen, and Dan Witzner Hansen. Low-cost gaze interaction: Ready to deliver the promises. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems*, CHI EA

'09, page 4453–4458, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605582474. doi: 10.1145/1520340.1520682. URL https://doi.org/10.1145/1520340.1520682.

Javier San Agustin, Henrik Skovsgaard, Emilie Mollenbach, Maria Barret, Martin Tall, Dan Witzner Hansen, and John Paulin Hansen. Evaluation of a low-cost open-source gaze tracker. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications*, ETRA '10, page 77–80, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781605589947. doi: 10.1145/1743666.1743685. URL https://doi.org/10.1145/1743666.1743685.

M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *Trans. Sig. Proc.*, 45(11):2673–2681, November 1997. ISSN 1053-587X. doi: 10.1109/78.650093. URL https://doi.org/10.1109/78.650093.

Sara Sereno, Patrick O'Donnell, and Keith Rayner. Eye movements and lexical ambiguity resolution: Investigating the subordinate-bias effect. *Journal of experimental psychology. Human perception and performance*, 32:335–50, 05 2006. doi: 10.1037/0096-1523.32.2.335.

Anders Søgaard. Evaluating word embeddings with fMRI and eye-tracking. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 116–121, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-2521. URL https://www.aclweb.org/anthology/W16-2521.

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. Sequence labeling parsing by learning across representations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5350–5357, Florence, Italy, July 2019a. Association for Computational Linguistics. doi: 10.18653/v1/P19-1531. URL https://www.aclweb.org/anthology/P19-1531.

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. Towards making a dependency parser see. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1500–1506, Hong Kong, China, November 2019b. Association for Computational Linguistics. doi: 10.18653/v1/D19-1160. URL https://www.aclweb.org/anthology/D19-1160.

Michalina Strzyz, David Vilares, and Carlos Gómez-Rodríguez. Viable dependency parsing as sequence labeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 717–723, Minneapolis, Minnesota, June 2019c. Association for Computational Linguistics. doi: 10.18653/v1/N19-1077. URL https://www.aclweb.org/anthology/N19-1077.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling,

C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf.

Jun Suzuki, Sho Takase, Hidetaka Kamigaito, Makoto Morishita, and Masaaki Nagata. An empirical study of building a strong baseline for constituency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 612–618, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2097. URL https://www.aclweb.org/anthology/P18-2097.

William R. Uttal and Pamela Smith. Recognition of alphabetic characters during voluntary eye movements. *Perception & Psychophysics*, 3(4):257–264, Jul 1968. ISSN 1532-5962. doi: 10.3758/BF03212741. URL https://doi.org/10.3758/BF03212741.

David Vilares, Mostafa Abdou, and Anders Søgaard. Better, faster, stronger sequence tagging constituent parsers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3372–3383, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1341. URL https://www.aclweb.org/anthology/N19-1341.

Xin Wang, Yuanchao Liu, Chengjie Sun, Baoxun Wang, and Xiaolong Wang. Predicting polarities of tweets by composing word embeddings with long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1343–1353, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1130. URL https://www.aclweb.org/anthology/P15-1130.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. Structured training for neural network transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 323–333, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1032. URL https://www.aclweb.org/anthology/P15-1032.

Rihana S. Williams and Robin K. Morris. Eye movements, word familiarity, and vocabulary acquisition. *European Journal of Cognitive Psychology*, 16(1-2):312–339, 2004. doi: 10.1080/09541440340000196. URL https://doi.org/10.1080/09541440340000196.

Erroll Wood and Andreas Bulling. Eyetab: Model-based gaze estimation on unmodified tablet computers. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, ETRA '14, page 207–210, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450327510. doi: 10.1145/2578153.2578185. URL https://doi.org/10.1145/2578153.2578185.

Jie Yang and Yue Zhang. Ncrf++: An open-source neural sequence labeling toolkit. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018. URL http://aclweb.org/anthology/P18-4013.

Daniel H. Younger. Recognition and parsing of context-free languages in time n3. *Information and Control*, 10(2):189 – 208, 1967. ISSN 0019-9958. doi: https://doi.org/10.1016/S0019-9958(67)80007-X. URL http://www.sciencedirect.com/science/article/pii/S001999586780007X.

Daniel Zeman and Philip Resnik. Cross-language parser adaptation between related languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*, 2008. URL https://www.aclweb.org/anthology/I08-3008.

Junru Zhou and Hai Zhao. Head-driven phrase structure grammar parsing on Penn treebank. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2396–2408, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1230. URL https://www.aclweb.org/anthology/P19-1230.

List of Figures

List of Tables

Listings

List of Abbreviations

bi-LSTM bi-directional long short term memory.

CRF conditional random fields.

EEG electroencephalography.

fMRI functional magnetic resonance imaging.

GECO Ghent eye-tracking corpus.

LAS labelled attachment score.

ms milli-seconds.

MTL multi-task learning.

NER named entity recognition.

NLP natural language processing.

POS part-of-speech.

UAS unlabelled attachment score.

UD Universal Dependencies.