Universität Bremen

Assessing the Emergence of Syntactic Information in the Sentence Gestalt Model During Sentence Comprehension

Sertug Gürel

(3216808)

Thesis submitted for the degree of: M.Sc. in Neurosciences

First Examiner: Prof. Dr. Thorsten Fehr

Second Examiner: Prof. Dr. Milena Rabovsky

Universität Bremen

21.09.2022

Universität Bremen

The following declarations are to be included in every exemplar of the Bachelor's and Master's Thesis

Name: Sertug Gürel Enrolment number: 3216808

Declaration of copyright

Hereby I declare that my Master's Thesis was written without external support and that I did not use any other sources and auxiliary means than those quoted. All statements which are literally or analogously taken from other publications have been identified as quotations.

Date: 21.09.2022 Signature S. Gürel

Declaration with regard to publishing theses

Two years after the final degree, the thesis will be submitted to the University of Bremen archive and stored there permanently.

Storage includes:

1) Master's Theses with a local or regional reference, as well as 10 % of all theses from every subject and year
2) Bachelor's Theses: The first and last Bachelor degrees from every subject and year

☒ I agree that for research purposes third parties can look into my thesis stored in the University archive.

☐ I agree that for research purposes third parties can look into my thesis stored in the University archive after a period of 30 years (in line with §7 para. 2 BremArchivG).

☐ I do not agree that for research purposes third parties can look into my thesis stored in the University archive.

Date: 21.09.2022 Signature S. Gürel

Contents

1. Abstract

Understanding and producing language can be considered an essential skill of the human brain. In machine learning, scientists create models that can imitate some aspects of this skill. The Sentence Gestalt model is a cognitively motivated artificial neural network for sentence comprehension. It is trained to predict semantic roles of the parts of sentences and does not use any syntactic information to perform this task. Also, few studies demonstrate that this model can capture some critical aspects of human language processing. From linguistic studies, we also know that implicitly learning syntactic information is an ability of humans. To the best of our knowledge, even though some recent studies focus on implicit learning of syntactic information in neural network models, none of these studies were done on the Sentence Gestalt model. In this thesis, we will investigate the question of whether or not the Sentence Gestalt model implicitly processes syntactic information. We used syntactic dependency relations to extract some syntactic features and the Sentence Gestalt model's internal representations to perform probing experiments. Our results indicate that the Sentence Gestalt model might be considering some syntactic information in its internal representations.

2. Acknowledgments

I would like to thank Prof. Dr. Milena Rabovsky to given me the opportunity to work on such a good research topic and for her supervision along the ups and downs of this thesis process. I also thank Prof. Dr. Thorsten Fehr for accepting to be an examiner for this thesis topic. I owe a big thanks to Dr. Alessandro Lopopolo for his invaluable guidance throughout all stages of my thesis project.

On a more personal level, I would like to express my gratitude to my parents for supporting me and my dreams. Another special thanks would go to Sergen Akaysoy for his support, encouragement, patience, and inspiration, even when there were thousands of kilometers between us. Nothing would be the same without you in my life.

I also would like to thank some friends, starting with B. Bahar Balcı. Her support for my mental health through international video calls was invaluable. I also thank Şule Taşlıyurt Çelebi and Eda B. Belek for their great friendship throughout all these times. Lastly, I would like to thank my friends from Bremen, Derya Tımartaş, Mira Sleiman, Nada El Jundi, and Stefano Masserini, for their companionship through this challenging road.

3. Introduction

Our brains are the result of evolutionary processes that took millions of years. Throughout this time, the brain not only increased in size but also became more structurally complex (Vallender et al., 2008). Indeed even compared to our closest non-human primate relatives, our brains differ in several ways, such as connectivity. As a result of this increase in complexity, humans gained a lot of complex cognitive abilities, including language. Some studies theorize that enriched connectivity in areas associated with language might be why humans developed more complex communication abilities than other primates (Ardesch et al., 2019). Naturally, this complexity is intriguing to our human curiosity, and it has become the quest of some neuroscientists to understand this complex and dynamic biological system. To this end, we investigate the brain from the molecular to the systemic level and try to comprehend the phenomena occurring during information processing. We use a variety of techniques for this investigation, from behavioral experiments to mathematical modelings.

Among all the cognitive processes scholars study, language is one of the interesting ones. The capability to produce and comprehend language is a unique trait of humans. Of course, this should not be interpreted as humans being the only species that can communicate. However, even by simple observation, it is evident that human language is more complex than the vocalizations of other animals used for communication. Therefore, researchers study many questions about language, such as 'How do we learn the language?', 'How do we produce it?' 'How do we understand it?', 'Which part of the brain is responsible for all these processes?'. These questions are not easy to answer, and demystifying these requires a great deal of work and effort.

Understanding the language is not the only thing we are trying to achieve. We also want to apply our findings practically. For example, some researchers are interested in the question of 'How can we implement some language related abilities of the brain to machines?'. Even though

one might think terms like machine learning and deep neural networks have emerged in the last ten years, the desire to implement some brain functions in computers is not new. In fact, history tells us that studies for these applications began between the late 1950s and early 1960s (Fradkov, 2020). Creating computer programs that can learn how to comprehend or produce human language has countless applicational potential. But it also opens doors to new questions for scientists, such as 'How do these learning algorithms achieve what they accomplish?' or ‘Can we learn anything from these models about language and brain?’.

All these questions about the brain, language, and machines bring us to our current study, which in broad terms it is focussing on language comprehension in artificial neural networks (ANN). The model we used in this project is called Sentence Gestalt (SG) model, a known ANN used for sentence comprehension (McClelland et al., 1989). It takes the words of the sentence and creates a probabilistic representation of the action that occurs in the statement. Moreover, the model's training is not considering explicit syntactic information. Nevertheless, some studies present strong evidence about the model’s ability to pick up critical elements of language comprehension. As a result of these studies, one can become curious about what else the model takes account of in its internal states.

In the literature, some recent studies on other neural language models focus on whether or not ANN models can implicitly learn syntactic information (Arps et al., 2022). This is an interesting question because, in linguistics, it is known that humans can implicitly learn syntactic relations (Rebuschat & Williams, 2009). So naturally, this makes us wonder if the SG model's inner states could also have some representation of syntactic information, even though the model is not trained on this type of data. If the model is already good at providing insight into human language comprehension, implicitly learning about syntax might be another aspect the model can grasp.

Due to the above reasoning, the aim of this study is to assess if the SG model is processing syntactic information implicitly. Answering this question will help to understand more about the SG model and sentence comprehension. As syntactic information, we will focus on dependency relations between words. First, some *syntactic measures* will be defined and extracted from sentences. Then, the internal representation of the SG model for the same sentences will be extracted from the model. Finally, a method called *probing classifiers* will be used to analyze the internal representations to see if there is any consideration of syntactic information. The following section will first explain the background of related terms and studies in detail so an understanding of the method and the results sections can be built from the ground.

4. Background

4.1. Language and Language Processing in the Brain

4.1.1. Language

The literal dictionary definition of language is stated as a structure used for written or verbal communication (Oxford University Press (a), n.d., para. 1a). In the context of science, the definition might vary according to the perspective of the particular scientific field. However, there are some common grounds majority can agree on it. Language is considered to have two main parts, one is the lexicon, and the other is the grammar. Simply describing the lexicon is a kind of dictionary composed of a language's words. Grammar is the set of rules that tells us how to use these words to create messages (Traxler, 2012). There are different aspects of grammar. Morphology and syntax could be some examples of the rule sets that form grammar. Morphology consists of the rules of word structures, and syntax has the rules about the order of the words in the sentence (Friedenberg & Silverman, 2012). Since our research will focus on syntactic relations between words, we will mainly cover that part of the language here. Linguistics is a big area, and it would not be possible to cover all of it in one sit.

Computations that process all these language rules and structures quickly happen in the brain. Research suggests that humans instantly decipher the words of sentences when they read or hear them. Moreover, some experimental studies indicate that people anticipate the upcoming words' characteristics. For example, Van Berkum et al. (2005) inspected if readers and listeners can use the knowledge about the discourse of a story to predict a specific upcoming noun. To test if participants formed a prediction about this noun, they continued the stories with a gender marked adjective that was incongruent with the syntactic gender of the noun. These incongruent conditions created a positive deviation in ERP waves compared to congruent conditions. Moreover, the effect disappears when the same sentences are given without the story that provides a discourse. Another example of the predictive processes in the brain about

language could be the Frank et al. (2015) study, in which they found some correlation between the amplitude of the N400 ERP component and the word's surprisal effect (which is a measure of word's predictability). In other words, we do not wait until the end of the sentence to process and form some predictions about what is coming next while reading or listening. This phenomenon is not only limited to the semantics of words, but also we are computing syntactic structures word by word. According to researchers, this processing method requires us to update the information with each incoming word (Otten & Van Berkum, 2008; Pickering & van Gompel, 2010).

Many theories in the language processing field argue that syntax might have an essential role in both production and comprehension of language. In order to convey meaningful communication, these rules must be learned. Humans start to form syntactic rules and relations in their minds at a relatively early age. Indeed, psycholinguistic studies show there is a connection between implicit learning and syntax acquisition, even in children of preschool age (Kidd, 2012). Of course, this is not limited to children. Also, adults who learn a new language can implicitly learn syntax. They form syntactic associations and learn aspects of structures by exposure without explicit awareness of the knowledge (Rebuschat & Williams, 2009; Petersson & Hagoort, 2012).

Even though doing all of these looks very easy for humans, the brain does plenty of work. Most of these tasks require higher brain functions and executive operations. Different aspects of the language must be integrated into each other to form holistic information. Also, if the brain analyzes language word by word and updates the meaning of sentences, one might hypothesize that the working memory should extensively involve in the process. Moreover, the brain has to deal with ambiguities and reference problems, if there are any (Pylkkänen & McElree, 2010). In the next section, we will review the neural organization of language in the brain and try to understand this complex structure.

4.1.2. Neural Basis of Language

Our mind uses language as means of creating a representation of objects and events in the outer world. It performs many computations not just on semantic or syntactic aspects of language but also on the relation of these aspects with other senses. Therefore, scholars hypothesized that language is a product of a relationship between three mental representations, phonological, syntactic, and semantic (also called *conceptual*) structures (see Figure 1). The relationship between phonological and semantic structure is mediated by syntactic structure.

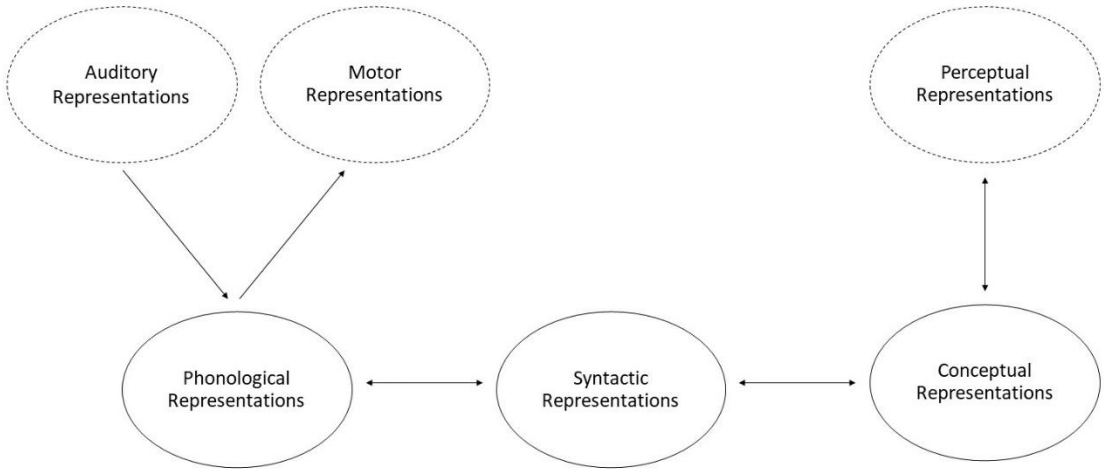

Figure 1. The visual portrayal of the relationship between mental representation of language is mentioned in the book chapter written by Jackendoff (2019).

The phonological structure is also connected with the auditory and motor systems so that people can comprehend and produce speech. Like phonological structure, the semantic structure has other connections too. It is connected with the perceptual system so people can talk about their observations about the environment (Jackendoff, 2019).

Due to these relations with other systems, language comprehension and production processes require many brain areas to work together. Regarding the organization of these areas, scientists propose a dual stream model similar to vision (see Figure 2). According to theory, there are two main information processing streams in the language processing system: ventral and dorsal streams. The regions in the superior and middle sections of the temporal lobe

contribute to the ventral stream, which is mainly associated with comprehension of language related information processing. The areas include Sylvian parietal temporal (Spt), and the posterior frontal lobe is considered in the dorsal stream, which is deemed to translate speech related brain signals to articulatory representations. Apart from these, researchers usually associate regions like the superior temporal gyrus (STG), superior temporal sulcus (STS), and anterior temporal lobe (ATL) (Hickok, 2009).

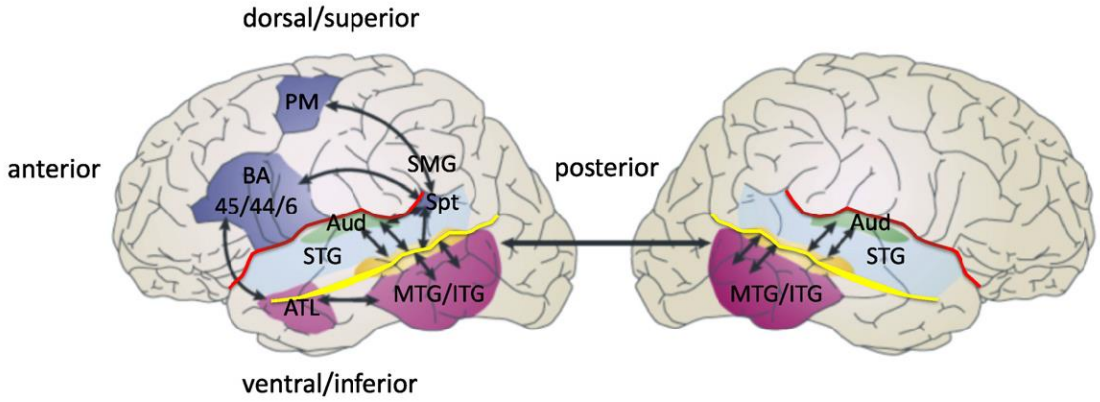

Figure 2. The dual stream model for language processing in the brain. (Taken from the paper written by Hickok (2009).) It can be seen that some brain areas related to language are bilateral, like MTG/ITG. Some other areas, such as BA 45/44/6, are unilateral.

In the previous part, we established how syntax is a critical aspect of language. However, it is hard to localize to one specific region when it comes to syntax processing in the brain. The scientific findings point out that interconnections between multiple regions are essential for processing syntactic information. Some studies claim that syntax is in the connections between brain regions (Kaan & Swaab, 2002; Petersson & Hagoort, 2012). When we examine the accumulated studies about the neural basis of syntax, we see that the posterior middle temporal gyrus is generally linked with comprehension and syntax production. The left inferior frontal gyrus (LIFG), posterior STG (pSTG), posterior middle temporal gyrus (pMTG), dorsal precentral gyrus (DpreC), and ventral precentral gyrus (VpreC) regions and their connections are related to syntax too. Also, some studies show that damage to connections between the

Broadmann 45 area and LIFG area causes problems with processing syntactic information (Pylkkänen, 2019; Matchin & Hickok, 2019).

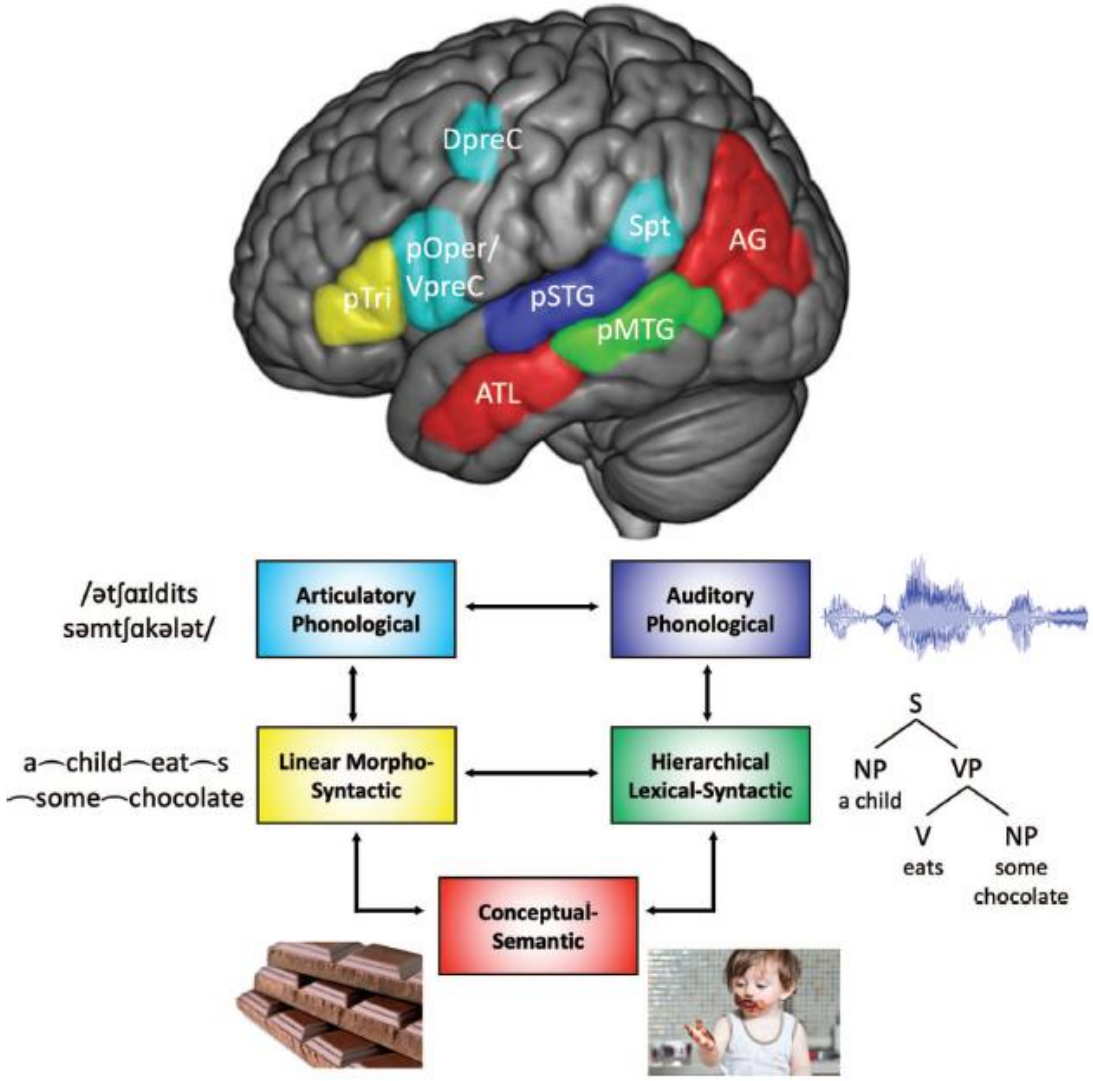

Figure 3. The brain regions are involved in syntactic processing with their assumed roles in language. (Taken from Matchin & Hickok's paper (2019).)

4.2. Machine Learning and Artificial Neural Networks (ANNs)

Making machines 'intelligent' is a long time passion in the computer science field. The main goal of the field is to produce computers and robots that can learn (Mitchell et al., 1990). The area is specifically concerned with making computers gather knowledge from data and later use this knowledge to make decisions or perform related novel tasks (Wang et al., 2009). From the neuroscience perspective goal of machine learning is divided into two parts. One goal

is to create new ways of analyzing neural data. The other goal is to use biologically and cognitively plausible machine learning models to develop theories about how the brain works (Hinton, 2011). Before proceeding further, it is essential to understand the fundamentals of machine learning. Therefore, this section will provide basic knowledge about machine learning and ANNs before answering more advanced questions about them.

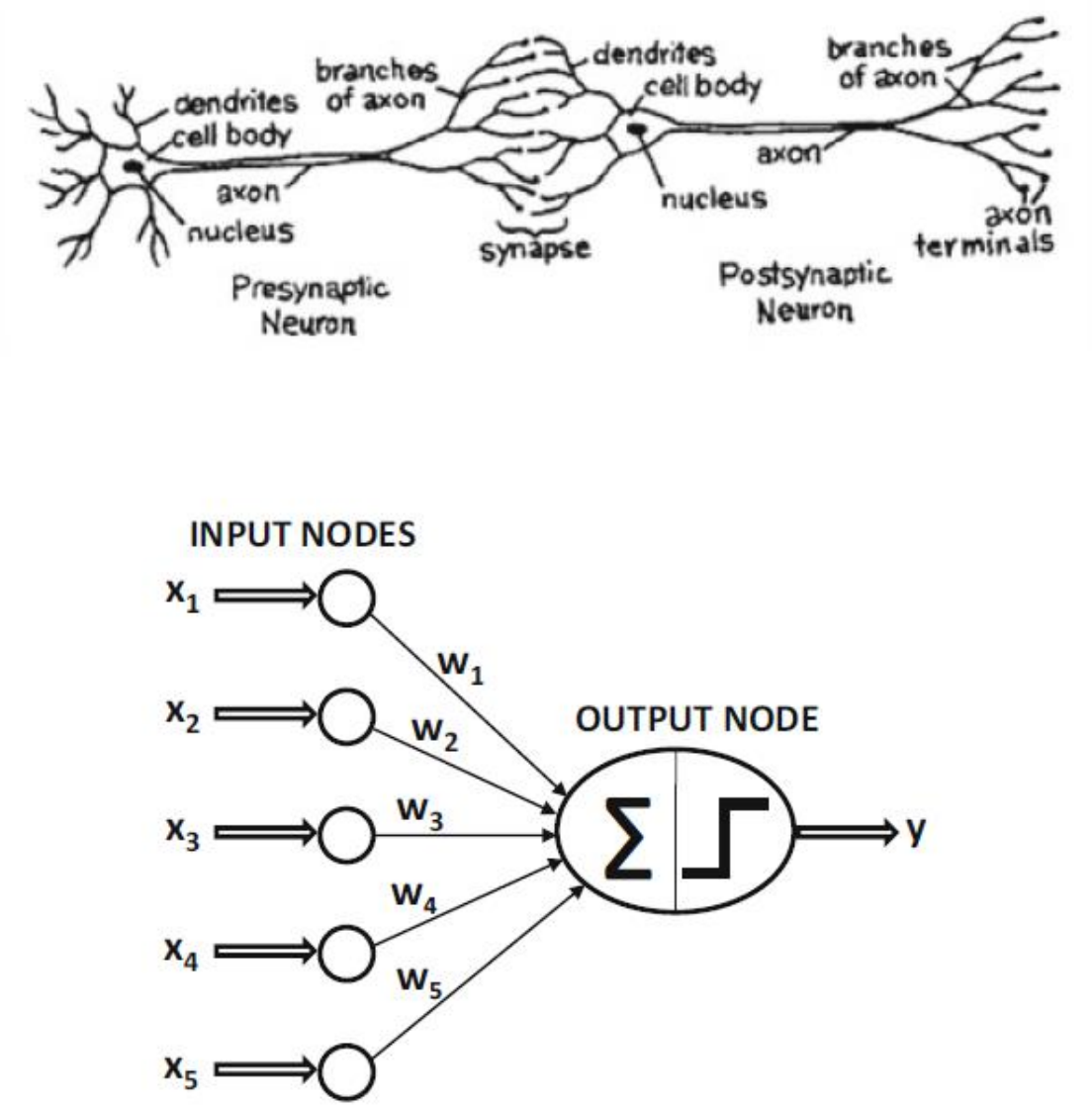

Figure 4. The representative drawing of actual neurons and the diagram of a single-layer artificial network. The inspiration for the architecture of artificial networks comes from real neurons. Each node in the artificial network can be imagined as neurons, and the relations defined by weights can be imagined as synaptic connections. (The figures are taken from the book chapter written by Aggarwal (2019a).)

One of the possible ways to create a machine learning model is to use artificial neural networks. In the last decade, these networks became popular, and today they are used in many fields, from engineering to medicine and even in the agriculture field (Abiodun et al., 2018). These networks are composed of layers of nodes connected to each other like neurons (see Figure 4).

The most simple neural network can be constructed as a single-layer network that could perform binary classification. As the name suggests, this type of network has one layer that input data enters and a connection to an output node. The number of input layer nodes depends on the amount of data features. The input data is defined as a vector such as $\bar{X} = [x_1, x_2, \dots x_k]$. The k represents the number of data features (Aggarwal, 2019a). For example, let us assume we want to classify daisies and sunflowers using petal size and center diameter of flowers. In this case, petal size and center diameter would be the two features of our data, and we would need two nodes in the input layer. Of course, this data is not the only thing we would require for this classification. We also need something to help us decide whether our classification result is correct. This would be our label vector that shows the actual target values for each sample in our data, which can be defined as $y_{observed} = [0, 1, 1, 0, \dots]$. Since it is a binary classification, we can categorize the vales as 0 and 1 and compare these actual values to our model's predicted values (Skansi, 2018).

Another component of neural networks is the weights. Weights define the connections between nodes, and like input data, it is also a vector that can be described as $\bar{W} = [w_1, w_2, \dots w_k]$. So each feature in the input data multiplies with corresponding weights, and the summation of these multiplications goes as an input to the next layer, which is the output node in a single-layer network. The output node passes this input through a so-called activation function (Mallot, 2015). For example, in the case of a binary classification problem, this function could be as follows;

$$activation\ function(a) = \begin{cases} 1, if\ a > 0 \\ 0, otherwise \end{cases} \quad (1)$$

This function would map the inputs as 1 or 0 that come to the output layer and make its prediction. We can mathematically summarize this process as follows;

$$\bar{W}\bar{X} = \sum_{i=1}^{k} w_i . x_i \quad (2)$$

$$y_{predict} = activaiton\ function\{\bar{W}\bar{X}\} \quad (3)$$

There are different types of activation functions that can be used for different types of models and purposes (Aggarwal, 2019a).

But of course, these are not enough to make a machine learn. We need something to tell our network how it performs. This is called loss function. Like the activation function, multiple functions, such as least square error, can be used as loss functions.

$$L = \sum (y_{observed} - y_{predict})^2 \quad (4)$$

The network can try to minimize this function by adjusting the weights and searching for the best possible weights that cause the minimum error (Anderson, 2014). The network can update the weights accordingly to the following function;

$$w_i^{new} = w_i^{old} + \alpha * (y_{observed} - y_{predict}) * x_i \quad (5)$$

$$\alpha = \begin{cases} +1, \ if\ answer\ is\ correct \\ -1, \ if\ answer\ is\ wrong \end{cases} \quad (6)$$

Besides single-layer network architecture, a multilayer approach is possible (see Figure 5). These networks have additional layers between input and output layers. Hence, they can be used for many applications, such as multiclass classification, and they can handle non-linearities in the data (Aggarwal, 2019a).

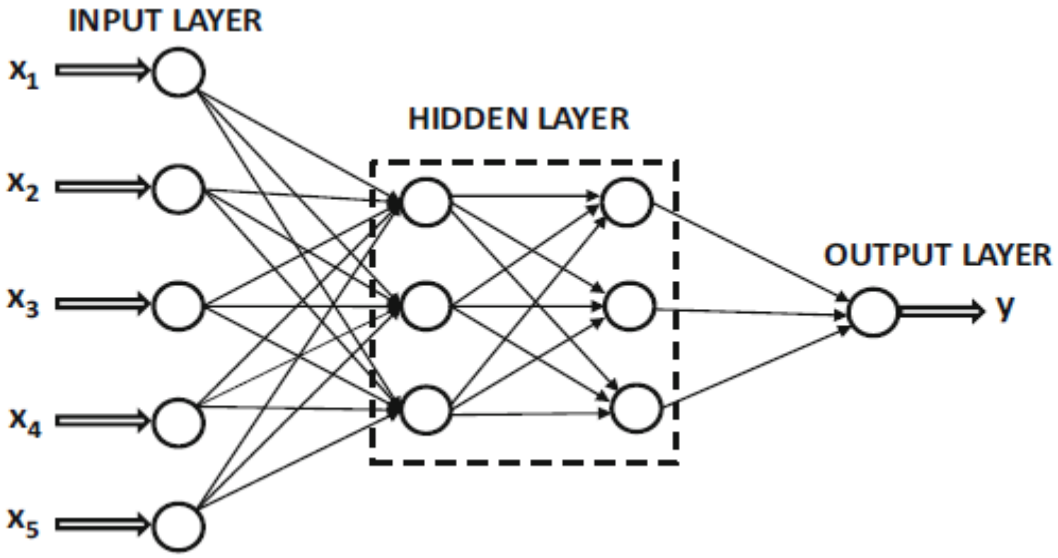

Figure 5. The architecture of multilayer artificial neural network. These networks can handle non-linearities and multiclass classifications. (The figure is taken from the book chapter written by Aggarwal (2019a).

After the training period, the model should be tested. In order to test the network, researchers usually divide their data into training and testing parts. After the neural network is trained by training data set, it can be tested by unseen testing data set. Creating multiple testing sets and performing the test process multiple times would also be beneficial to validating testing results. Besides dividing the data and using one part for training and the other for testing, researchers can also use a method called *cross-validation*. This method divides the data into *n* parts, also known as *n-folds*. Then one part would be separated for testing, and the *n-1* part can be used for training. The process repeats itself for *n* amount of time, and each time different parts are set aside for testing, the rest of the data can be used for training the model from the start. In the end, the average of this *n* number of the training-testing cycle can be taken (Liu,

2006; Aggarwal, 2019a). The main consensus on cross-validation is that it is a good way to evaluate a model, especially if there is a limited amount of data. This way, every part of the data will be used in training and testing, and the model's performance will be better estimated. The process can also be used when we do not know about data, and we need to select model parameters that would perform best with data. Parameters can be selected from a parameter pool, and we can test each parameter set with the cross-validation method.

4.2.1. Overfitting

Before ending this small introduction to the neural networks section, it would be beneficial to mention a problem that might occur in neural networks, which is called the overfitting problem. This problem means that the network is memorizing the training data. As a result, it can not generalize what it learned and performs poorly when tested by unseen test data (Aggarwal, 2019a; Bejani & Ghatee, 2021). This situation would cause wrong inferences from data and may mislead the scientist. Therefore, the researchers should be careful and use methods to minimize this risk. It is quite fortunate that the machine learning field has developed various methods to deal with this unwanted situation.

4.3. Probing Classifiers and Diagnostics in Neural Networks

It is probably impossible for someone working with artificial neural networks to have never heard of the saying that neural networks are like a *black box*. This approach to using neural network models is akin to behaviorism's approach to mind and brain in the early phases of cognitive sciences (Eliasmith, 2005). Even though we extensively use neural network models to perform some tasks today, there are still many questions about what they are learning and what is happening in their internal states to perform these tasks. Decoding the information processed in these models is difficult, and scientists still discuss the best way to do it. Especially in the natural language processing (NLP) field, researchers want to understand how cognitive computational models learn the syntax, subject-verb agreement, or semantic properties

(Giulianelli et al., 2018). Many of them focus on developing and perfecting methods to inspect these properties.

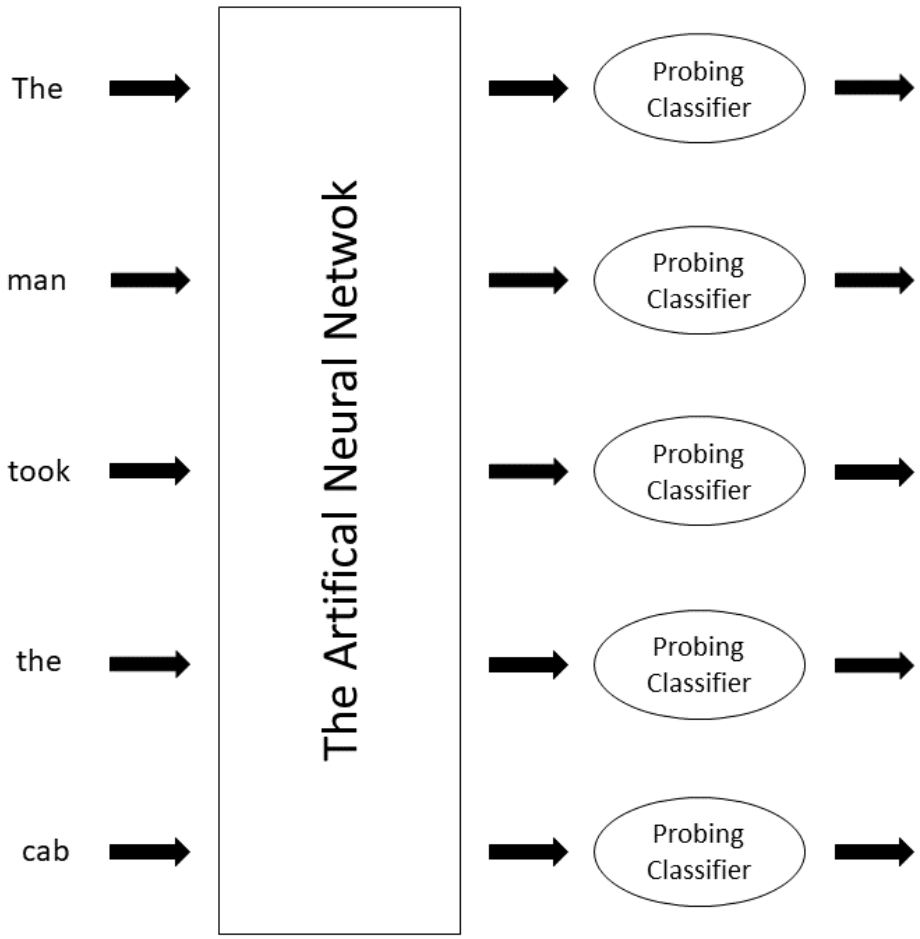

Figure 6. An exemplary figure for the process of probing. The sentence enters the artificial neural network word by word. Then, internal representations of words are taken from the model and passed through a probing classifier to train the classifier for a particular language property we are searching for.

To understand the internal representations of these models, the usage of one method gained popularity in relatively recent years. It is called the probing classifiers. Basically, the idea is to take the internal representations of a neural network occurring during information processing and use them for training a classifier to perform a task about the property we want to inspect. If the classifier manages to perform the task or learns the information related to the task from internal representations, then this might mean that the neural network under scrutiny considers this property during its information processing (Alain & Bengio, 2016; Belinkov, 2022). The

first publications about this method can be traced back six or seven years, so it is not a very old method. Therefore, there is still much discussion about it. Some of these discussions revolve around fundamental questions such as ‘how to evaluate the results of these classifiers?’ and ‘what type of classifiers are suitable for what type of questions?’.

Probing studies show that neural network models in the NLP area can represent much information in their internal representations. In their study, Conneau et al. (2018) designed ten probing experiments to examine which features of the sentences were processed by a NLP model. For example, *sentence length* is one of the features they chose to do a probing experiment. First, they group the length of each sentence into six equal length range groups. Therefore, they created a multiclassification problem with six classes. Then they conducted the probing experiment to see if the probe could predict the sentences’ length group from the internal representation of their model. Finally, they reported the probing accuracy with some comparison metrics to evaluate, such as baseline accuracy. This baseline accuracy can be calculated by assuming the model does not learn but randomly guesses a class or always guesses the most common class label. Later it can be compared to the actual task accuracy that the classifier achieved. Theoretically, if the classifier learns anything, their accuracy should be higher than this baseline accuracy. In the end, Conneau et al. (2018) reported that their probe’s accuracy for sentence length was better than the baseline accuracy for sentence length. This could indicate that their NLP model is processing information about sentence length in its internal representations. However, some researchers claim that reporting only accuracy might be misleading, especially when there is a class imbalance. Therefore, some choose to evaluate the classifier's performance in other ways, like reporting a confusion matrix and f1 scores or proposing to design a comparison task. For example, Hewitt and Liang (2019) worked on developing this type of task by linking words with random class labels so that the classifier

would be less selective if it is learning the actual target property. However, there is no clear consensus on this matter at the moment (Belinkov, 2022).

The choice of the classifier is another issue and an ongoing debate in the probing classifiers subject. The scientists are mainly divided into two groups: the ones say it should be a simple linear classifier, and the others say it should be a complex, multilayered, and non-linear classifier. There are plenty of studies that use either type of classifier and can achieve some level of satisfying results. Other than these issues, the scientists also point out that the probing experiments only tell us if a particular property is represented in the artificial neural network or not. The question of if this property is used to make predictions is another research topic (Belinkov, 2022).

4.4. Sentence Gestalt (SG) Model

In the beginning, it has been stated that this study will focus on the SG model to investigate if there is emergent syntactic information in the model's internal representations. Therefore, it would be wise to dedicate a section to the model. This section will try to shed light on the theoretical foundation of the model, how it works, and some examples of additional studies conducted with it.

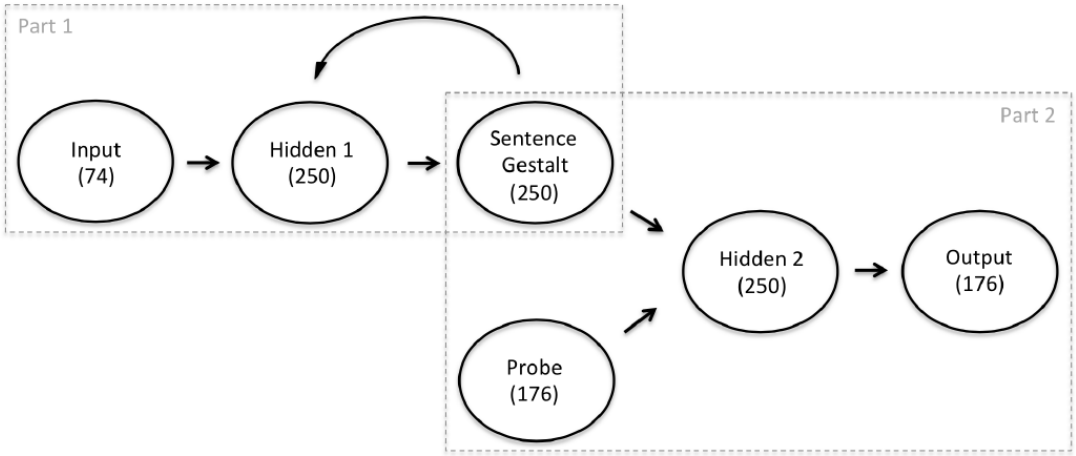

Figure 7. The diagram of the sentence gestalt model (taken from Rabovky et al., 2016). Part 1 corresponds to the update network. Part 2 is the query network. The probe in the query network should not be confused with probing classifiers. This probe part probes the neural network for thematic roles like the agent and object.

The SG model is an artificial neural network developed for sentence comprehension (McClelland et al., 1989). The studies performed with this model show that this neural network is able to grab some features of human language comprehension successfully. The theoretical working principle of the model relies on the knowledge actually mentioned in the language processing section of the thesis, which is that humans process sentences immediately without waiting for the whole sentence to finish. With each word given, they update their representation of events mentioned in the sentence. They even have some anticipations and predictions about incoming words (Otten & Van Berkum, 2008; Pickering & van Gompel, 2010). Similar to this, the SG model also processes sentences word by word and revises its internal representations of the event described in the sentence. An essential feature of the model is that it does not use explicit syntactic information to do this or does not have any separate part that processes syntax.

The model has two main parts, one of which is called the *update network*. During the training of the model, input enters from this part, and every incoming word updates the SG layer. Then, it is directed to the other part, called the *query network*, where the model gives some questions (which are called probes) concerning the events in the sentence. These questions are about the predicate, patient, and agent in the sentence. This probing with questions process takes place after each word. Through this questioning, the model is trained. The error occurs from the comparison of the model's predictions and actual answers to questions used to adjust the stochastic gradient descent. As a result, the model learns to comprehend sentences by correctly distinguishing these parts.

It is important to question how much parallelity this model has with human cognition. Some researchers that work with the model are trying to reveal whether or not the model captures the other aspects of language comprehension. One particular study topic worth mentioning is the one that focuses on the relation between the SG model and the N400 ERP component that occurs in human sentence comprehension. N400 component in the language processing field is

linked with comprehending the meaning of the incoming stimulus. Research shows that when people are presented with a word that does not fit the context of the sentence, the brain generates a larger N400. Therefore, some scientists theorize that the N400 component indicates the update process that occurs due to added new information in the mental representation of sentence meaning. Since the SG network does a similar updating during sentence comprehension, they develop a mathematical model for the N400 component out of the SG layer's update (which they refer to as the semantic update) (Rabovky et al., 2016, 2018). Later in another research, they successfully used this modeling to predict N400 amplitudes of people who do a naturalistic sentence reading task (Lopopolo & Rabovsky, 2021). This is an interesting accomplishment on behalf of the model. These findings indicate that the SG model is an adequate neural network that can offer some insights into the sentence comprehension process. Therefore, this makes us think it could have more properties similar to human sentence comprehension, such as implicit syntax learning. This thesis will be the first step to finding answers to this question. In the following section, we will focus on explaining the data, the methods, and the overall procedure we used in the study.

5. Methods

In the introduction, we briefly covered what type of method we can adopt to inspect the internal representations of a neural network model. Later, in the background, we gave some information about probing classifiers and the SG model. This section will deepen readers' knowledge of our data and methods. First, we will discuss the corpus, which is the data that the SG model trained and used for developing the *syntactic measures* (the syntactic properties we want to search for). Then we will focus on what type of syntactic information is used and how syntactic measures are created. Finally, we will close this chapter by discussing which machine learning models are used as probing classifiers.

5.1. Data

5.1.1. The Corpus

The corpus in linguistics context means data from a collection of sentences linguists can use for analysis and research (Oxford University Press (b), n.d., para. 3b). In this study, our SG model previously trained by Lopopolo and Rabovsyk (2021). They used a corpus containing the sentences' semantic role labels called Rollenwechsel-English corpus (RW-eng) (Sayeed et al., 2018), which is based on ukWaC (Ferraresi et al., 2008) and British National Corpus (BNC Consortium, 2007). The role labeling process for the corpus was done by a role labeler SENNA. A critical property of this role labeler is that it does not rely on syntactic parsing of the sentence to understand the relationships. Therefore, the syntactic information would not play a confounding role in semantic role labeling (Sayeed et al., 2018).

5.1.2. Training of SG Model with the Corpus

As stated above, the SG model used in this study was pre-trained by Lopopolo and Rabovsyk (2021). They determined the parameters, and used the Adamax optimizer (Kingma & Ba, 2014) for the model. The learning rate of the optimizer was set to 0.0005. They also applied some filtering to the sentences. The word number of the sentences changed between 4

and 25, and the maximum number of frames was limited to 10. In the end, they trained the model with a little over 2 million sentences, and they used around 200 thousand unseen sentences for testing.

5.1.3. Dependency Grammar and Parsing

Other than semantic role labels, the corpus contains another type of information called dependency grammar (also known as dependency structure). These structures represent the syntactic information as dependency relations between words. The word that customizes or supplements another word is called *dependent*, and the other word is called *head*. Dependency structures also carry information about the functional nature of the relation. There are computer algorithms that are called dependency parsers. These parsers create these dependency structures according to various criteria. We used our corpus's Spacy dependency parsing data to create syntactic measures.

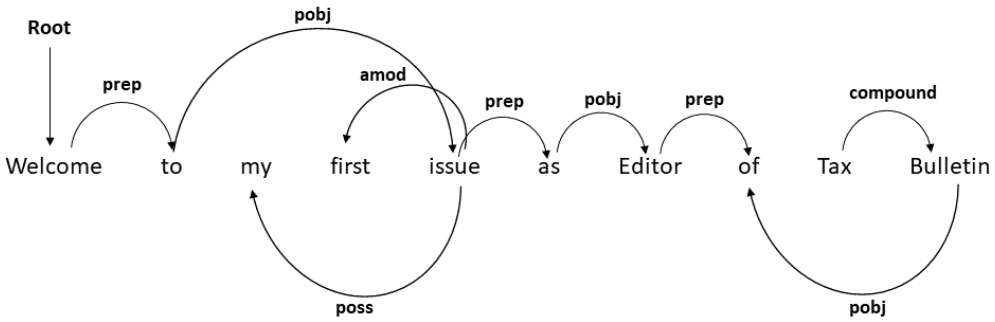

Figure 8. An example of the dependency structure. The blunt sides of the arrows are heads, and the sharp side of the arrows are dependents. There is also the root, a hypothetical construct that anchors the sentence.

5.2. Dependency Measures

In order to do a probing experiment on the SG model and see if there is any syntactic information implicitly processed by the model, we have to quantify the syntactic properties we want to check. Dependency structures are good representations to capture syntactic relations between words. Hence it could help develop some syntactic measures. There are some examples

in the literature about syntactic dependency measures. One recent example is Lopopolo et al. (2021) study about syntactic processing in the brain. From their example, we came up with six dependency measures. These measures would become labels for our probing experiment, meaning an observed value we can compare to the probes prediction.

5.2.1. "Is it head?" (is_hd) dependency measure

Since the dependency structures revolve around dependent and head relations between words, checking if the SG model processing this property in its internal representations might be a good start. As one might infer from Figure 8, in the dependency structures, all words are dependents of some word, but not all words are the head of another word, and some words can be the head of multiple words. This gives us a chance to quantify this information in a binary fashion. If a word is not a head, we can label it as 0. If a word is a head, we can label it as 1. This would be an excellent syntactic property measure for use with a binary classifier.

Table 1

An Exemplary Exhibit of the is_hd Measure

	Welcome	to	my	first	issue	as	Editor	of	Tax	Bulletin
is_hd	1	1	0	0	1	1	1	1	0	1

5.2.2. "Left-hand side heads" (lhd) dependency measure

The results of N400 studies with the SG model gave us reasons to assume that model has some expectations about incoming words and their roles. Therefore, dividing the 'is it head' measure as left and right-hand sides would be an interesting approach. In the left-hand side heads measure, we only count the heads on the left side of the dependents. Therefore, it would be another measure we can use with a binary classifier. Since the model sees the left-hand side

heads before the dependent, if it is processing this type of connection, it will not be surprising to see the probe being able to learn this syntactic property.

Table 2

An Exemplary Exhibit of the lhd measure

	Welcome	to	my	first	issue	as	Editor	of	Tax	Bulletin
lhd	0	1	0	0	1	1	1	1	0	1

5.2.3. "Right-hand side heads" (rhd) dependency measure

Table 3

An Exemplary Exhibit of the rhd measure

	Welcome	to	my	first	issue	as	Editor	of	Tax	Bulletin
rhd	0	0	1	1	0	0	0	0	1	0

The right-hand side heads will be the opposite of the left-hand side measure. Here, we will count only the heads on the dependent's right. As stated above, it will not be surprising if the model processes the heads on the left. However, the right side heads are the incoming dependency connections, so the model process the dependent before processing the right-hand side head. Therefore, it would be interesting to see if the model has any anticipation of these incoming syntactic connections.

5.2.4. "Number of dependencies" (nr_dps) dependency measure

Using word countings could be another way of creating measures. This is the type of dependency measure used in the Lopopolo et al. (2021) study. In this measure, we will count

the number of dependency relations each word has in the sentence. This type of dependency measure would be suitable for a multiclass classifier.

Table 4

An Exemplary Exhibit of the nr_dps measure

	Welcome	to	my	first	issue	as	Editor	of	Tax	Bulletin
nr_dps	1	2	1	1	4	2	2	2	1	2

5.2.5. "Number of left-hand side dependencies" (nr_ldps) dependency measure

Similarly to a "is it head?" measure, we can divide the number of dependency relations measure into left and right-hand side measures. The value of the words that do not have any left-hand side connections would be 0.

Table 5

An Exemplary Exhibit of the nr_ldps measure

	Welcome	to	my	first	issue	as	Editor	of	Tax	Bulletin
nr_ldps	0	1	0	0	3	1	1	1	0	2

5.2.6. "Number of right-hand side dependencies" (nr_rdps) dependency measures

Lastly, we have the number of right-hand side dependency relations measure. If a word has no right-hand side connection, its value will be 0. The primary purpose of this measure is similar to the right-hand side heads measure. It will be another good way of seeing if the SG model anticipates the number of incoming connections.

Table 6

An Exemplary Exhibit of the nr_rdps measure

	Welcome	to	my	first	issue	as	Editor	of	Tax	Bulletin
nr_rdps	1	1	1	1	1	1	1	1	1	0

Using the Python programming language version 3.7 (Van Rossum, 2009), we created algorithms that take the sentences and dependency parsing data from the corpus and create vectors for each sentence, as seen in the above tables.

5.3. Support Vector Classifier

We covered terms related to machine learning, simple binary classifiers, loss functions, and activation functions. Although previously we mainly focused on single-layer perceptron, it is not the only binary classifier. Various methods can be used for binary classification. These methods usually differ in some way, such as their activity or loss functions. This difference makes each of them have some strong and weak sides compared to the others. This variety puts researchers in a position where they have to decide on the machine learning model that suits their goal because the selection of the model can affect the results directly. In this research, we chose to use a support vector classifier (SVC) that belongs to a class of machine learning models known as support vector machines (SVM).

SVM is a supervised learning model developed by Cortes and Vapnik (1995) (see Figure 9). The main idea of SVM is to find the best hyperplane that separates the data into two classes. Of course, one can fit multiple hyperplanes that separate a data set. However, this does not mean every hyperplane selection results in good generalization. To ensure a satisfactory level of generalization, SVM adopts a soft margin approach. It aims to reduce generalization problems by finding the best-fitted hyperplane with the maximum margin between two classes.

The *support vectors* are called to vectors that are on the edge of the margins (Kotsiantis, 2007). The optimization process of this hyperplane and margin is as follows;

$$\begin{aligned} &minimize\ \tfrac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\varepsilon_i \\ &subject\ to\ y_i(w * x_i) - b = 0 \qquad (7) \\ &\varepsilon_i \geq 0 \end{aligned}$$

The mathematics of the SVM is not the main point of this master thesis. Therefore we are not going too deep into analyzing the above function. However, one of the critical parameters in this optimization process is the *C* value. This parameter is known as the regularization parameter and affects the size of the margin (Cherkassky & Ma, 2004).

The SVMs' loss function is the main distinction that separates this model from other machine learning models. SVM uses a loss function called the *hinge-loss function* (function 8).

$$L_i = max\{0, 1 - y_i^{observed} y_i^{predict}\} \qquad (8)$$

In this function, positive training prediction is penalized when they are less than 1, and negative training prediction is penalized when they are greater than -1. In other words, the model's predictions are not penalized when they are just different from the target value (see figure 10). Moreover, it is not penalizing values for being *too correct*, but it penalizes them for being *too wrong* (Aggarwal, 2019b).

There are a few upsides to using the SVM method. For example, it can be used for non-linear classification. This is made possible by a method called the *kernel trick*. It projects the vector data to a higher dimensional space and tries to separate them into two classes. As a result,

the hyperplane becomes a 2D sheet instead of a line in this higher dimensional space (see figure 11).

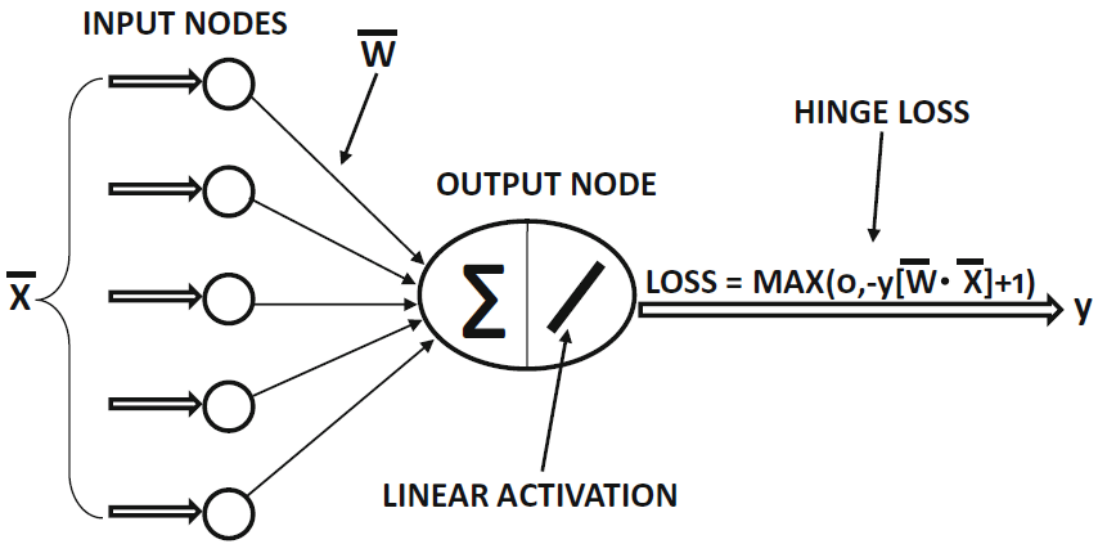

Figure 9. The basic architecture of the SVM model. (Taken from Aggarwal, 2019b)

Although it can handle non-linearities, the SVM is still inherently a binary classifier. One can also use tricks for multiclass classification, such as the *one versus one* (OVO) method. OVO takes all classes and makes binary classification between all possible combinations. However, it is not always a preferred future because training multiple models and making comparisons turn the method into a computationally expensive one. There are much better machine learning models for multiclass classification (Vapnik, 2000; Mahesh, 2020). Another advantage of SVM is that due to the nature of its optimization process, it avoids ending up at local minima, which is a problem with other methods like neural networks (Kotsiantis, 2007).

There are also a few downsides to the SVM method too. One example of these the researcher has to perform a hyperparameter tuning process to determine the best C value or kernel before doing the final training end test. This way, they can ensure they will have much better predictions (Kotsiantis, 2007). However, this is a computationally heavy process, especially if the sample size of the data is too big.

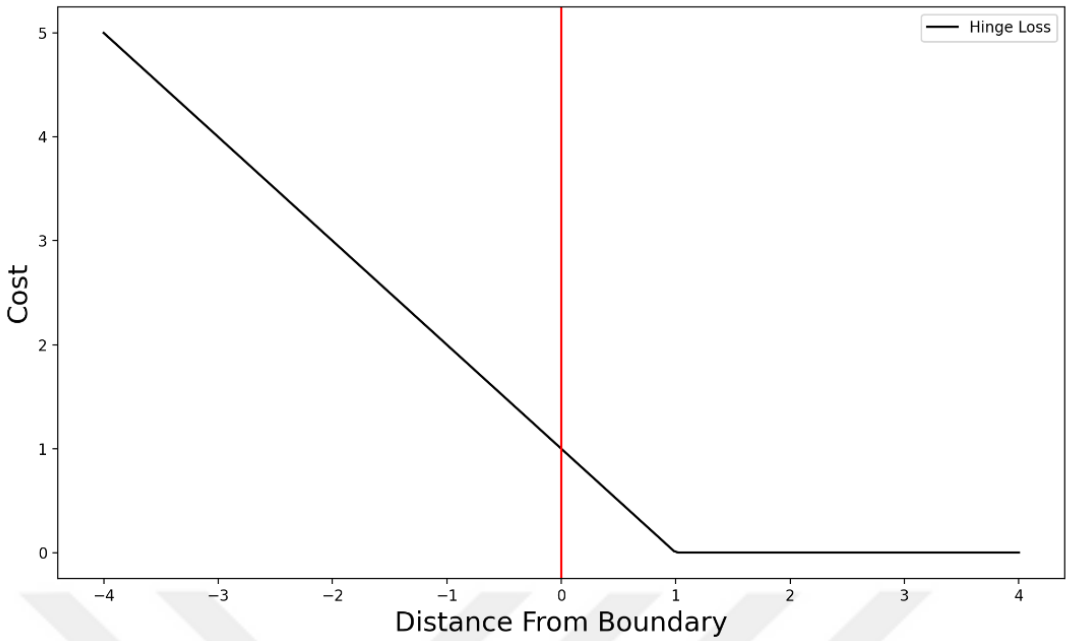

Figure 10. The plot of the Hinge-Loss function. As it can be seen here clearly, after 1, the function becomes flat, it does not penalize for being too correct, but it penalizes for being *too wrong*.

This study used SVC as the probing classifier to test if the SG model considers the binary syntactic measures. We considered this method a fine choice because it is a simple classifier, but at the same time, it can manage non-linearities, and its optimization process makes sure it will not be stuck in a local minimum point. We created our model using the Scikit-learn library's SVC function (Pedregosa, 2011). In order to tune the parameters of the model, we adopted a randomized search process with cross-validation. We trained the model with these randomly selected parameters over and over again. Ultimately, we selected the parameter set that returned the highest accuracy as our best parameters. After the parameter tuning process, the model is trained and tested with unseen data by cross-validation. The number of folds selected as five means that data was divided into five parts, and training was repeated five times. Each different time, a different part was the testing data. We produced the normalized confusion matrix, precision, recall, and f1 metrics for each cross-validation testing and got their average.

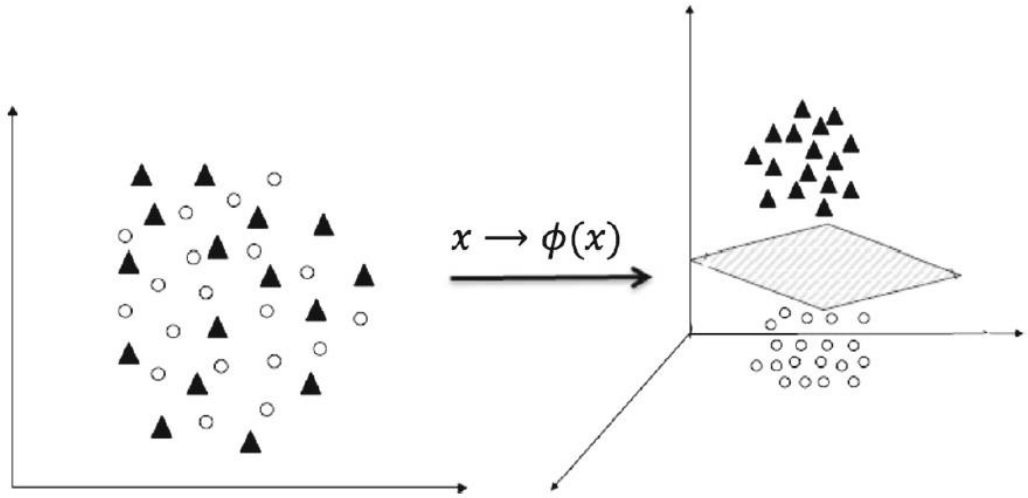

Figure 11. The figure shows the so-called kernel trick. In this example, data was projected from 2D to 3D space, making it possible to do a linear separation. (Taken from Chauhan et al., (2018)).

5.4. Multiclass Classification with Multilayer Perceptron

How can we do a probing experiment if we have more than two classes at hand? As one might imagine, not all problems we try to solve by machine learning are reducible to binary groups. What if we want to build a model that recognizes letters? For the English alphabet, that would make twenty-six classes. Luckily there are methods and algorithms that we can use to achieve this. In the introduction, we briefly mentioned multilayer perceptron (MLP). Here we will explain the usage of MLP in multiclass classification and how we build our second probe for multiclass syntactic measures.

For multiclass classification, we need slightly change the composition of the perceptron. In binary classification, we had an output layer containing only one node. The weighted sum that reaches the output node would go through an activation function that can map the input to either 1 or 0 to predict a class. In a multiclass classifier, we have more than one output node (see Figure 12). Therefore we get an output vector instead of a single number. There are different ways to compare this multi-output to our target value. For example, setting the output layer's activation function to *softmax* function (Function 9) (Aggarwal, 2019b).

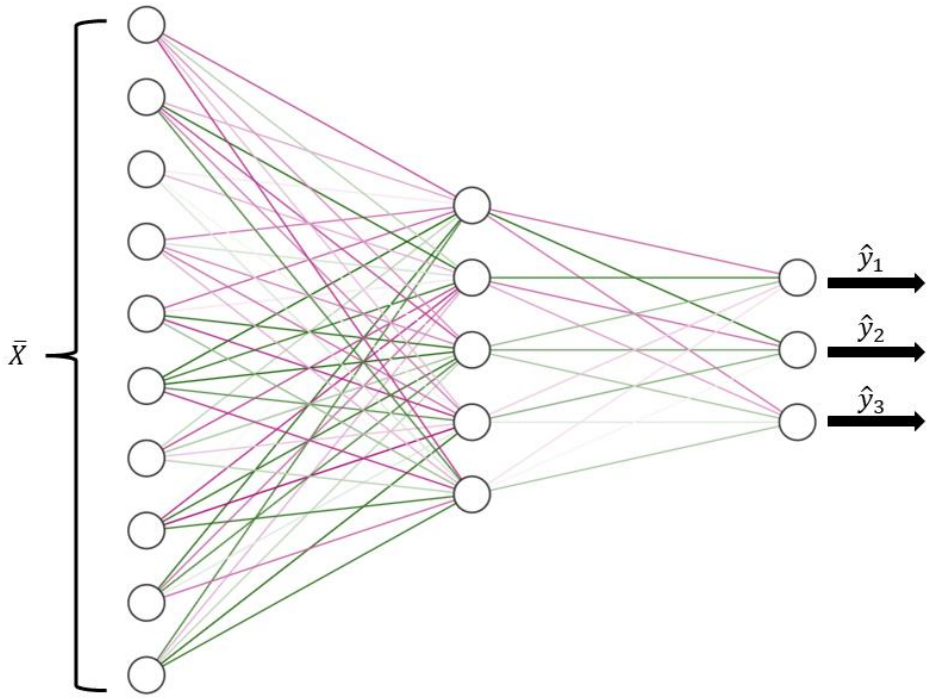

Figure 12. The basic architecture of multilayer multiclass network.

$$\sigma(z_j) = \frac{e^{z_j}}{\sum_{n=1}^{N} e^{z_n}}, \quad j = 1, \ldots, N \qquad (9)$$

The function transforms our output vector into a probabilities vector, and the class with the largest probability is selected as the predicted class (see Figure 13). Then one can calculate the loss and optimize a gradient descent to update weights via the backpropagation method. In the backpropagation method, the weights update happens from the output layer to the input layer (Han et al., 2012; Aggarwal, 2019a).

We constructed our second probe for our multiclass syntactic measures using the Scikit-learn library's MLPClassifier function (Pedregosa, 2011). Our probe consists of an input layer, a hidden layer, and an output layer. The activation function of the first two-layer is determined

as rectified linear unit (ReLU) function, and the output layer's activation function is the softmax function.

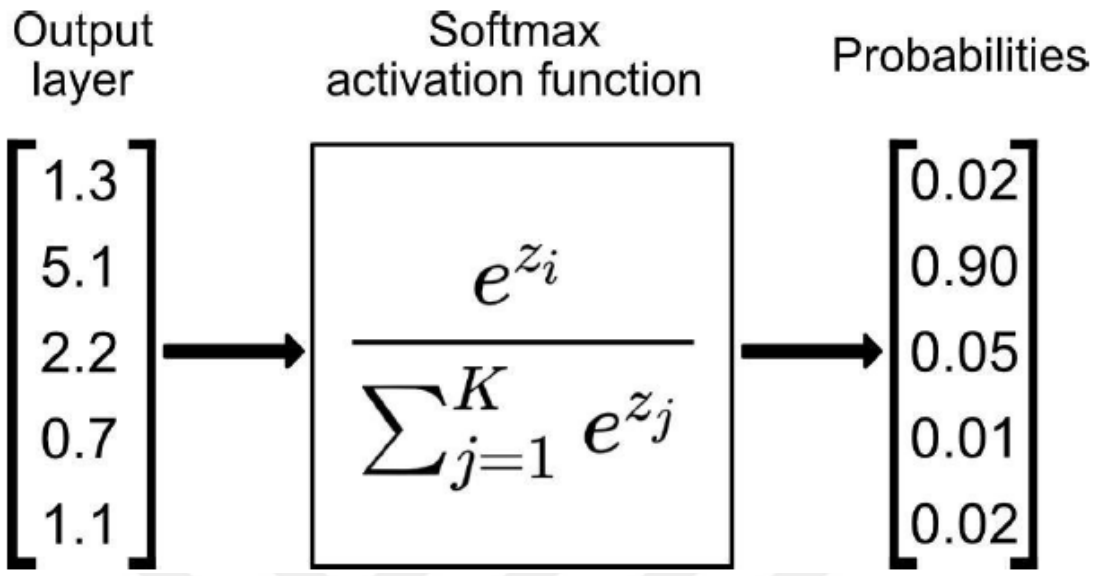

Figure 13. A representation of how the softmax function performs on the output layer. *i* is the index of the node. *z* is the value of the node (Nabiyev & Malekzadeh, 2021).

The ReLU activation is a widely used function in neural networks because it is a biologically inspired method (Szandała, 2020). This function does not allow the artificial neuron (node) to fire if the input equals zero or lower. However, if the function's input exceeds zero, it will be fired (see Function 10 and Figure 14). Firing in this context means the node will send an input to the next layer other than zero (Sharma et al., 2020).

The loss is calculated with the *Cross Entropy Loss* function. For gradient optimization, we used the *Adam* function (Kingma & Ba, 2014). The parameters for the Adam optimizer are determined as follows; the learning rate is 0.001, the beta-1 is 0.9, the beta-2 is 0.999, and the epsilon is 10^{-8} (Hewitt & Manning, 2019).

The model is trained by a cross-validation procedure and folds number selected as five like our binary SVC probe. Finally, the normalized confusion matrix, precision, recall, and f1 metrics were calculated and averaged.

$$f(x) = \max(0, x) \qquad (10)$$

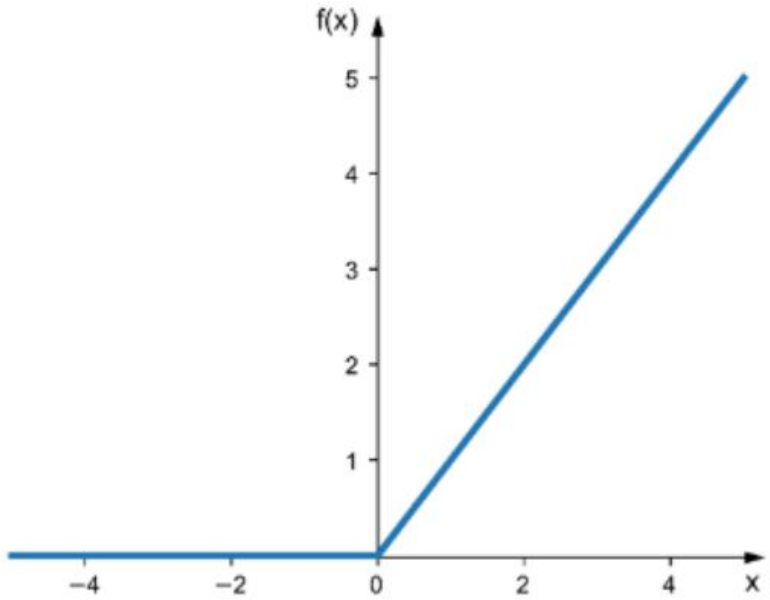

Figure 14. The graph of the ReLU function. It can be seen clearly that the function does not make the artificial neuron active if the input value is zero or less.

5.5. Experimental Procedure

Before we move on to the results section, we want to summarize the procedure of the study. First, we will handle the recording of internal representations of the SG model. These internal representations will be obtained from the Sentence Gestalt layer of the model. We will filter the sentences before they go through the model. The selected sentences' word length will be between 4 and 15, and the maximum number of frames will be 2. The model will process the sentences from the corpus word by word, and the internal representations will be saved for each word. Then our dependency measure algorithms will go through the same sentences and produce target labels for each word. Using these internal representations and the dependency measures, we will perform our experiments with probing classifiers. In order to train the probes, the internal representations of the SG model will pass through it, and the predictions of the

probe will be compared to dependency measures. At the end of the process, we will compute confusion matrices and some metrics, such as the f1 score. These would be enough to evaluate the probes' performance and see if the SG model processes syntactic information in its internal representations.

6. Results

6.1. Internal Representations and Dependency Measures of Words

First, our algorithm reads the sentences from the corpus. Then sentences were filtered according to the criteria we determined, such as the length of the sentence and the maximum number of frames. Then, each sentence was fed to the SG model. For each word, the internal representations from the gestalt layer of the model were recorded. Finally, six dependency measures mentioned in the method chapter were created for each sentence. These measures are as follows; 'is it head', 'left-hand side heads', 'right-hand side heads', 'number of dependency relations', 'number of left-hand side dependency relations', and 'number of right-hand side dependency relations'. As a result of this process, we end up reading and creating dependency measures for almost 39 thousand sentences. These sentences had little more than 348 thousand words in total. Before moving into probing classifications, we checked some descriptive statistics for our dependency measures, such as class distributions.

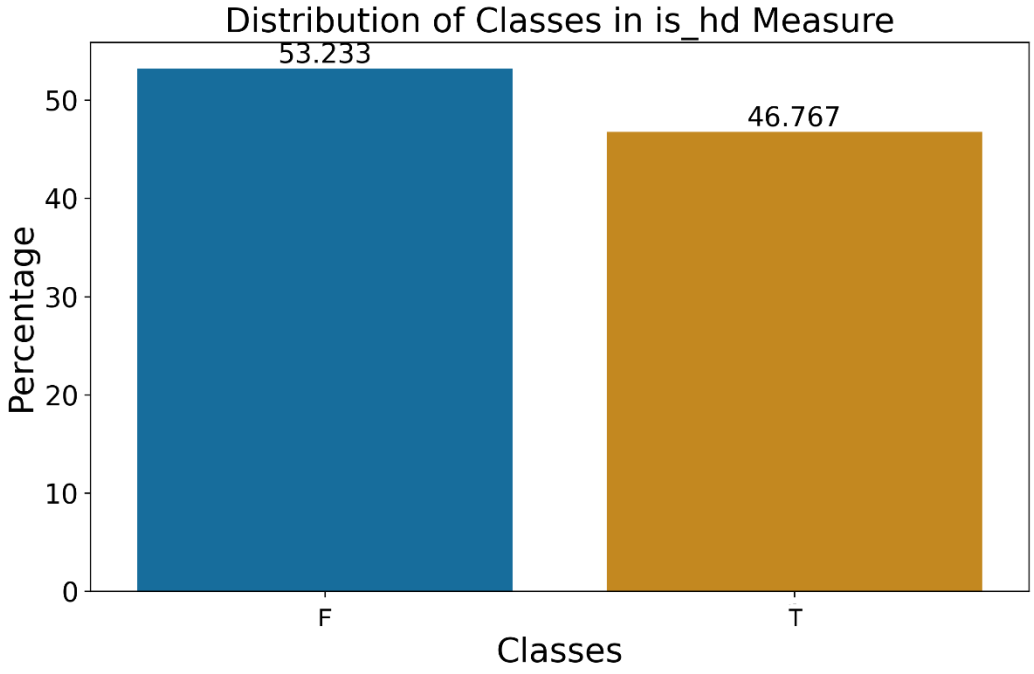

Figure 15. The class distribution of "is it head?". The F stands for False and represents non-head words. The T stands for True and represents the head words. It shows a slight imbalance between classes.

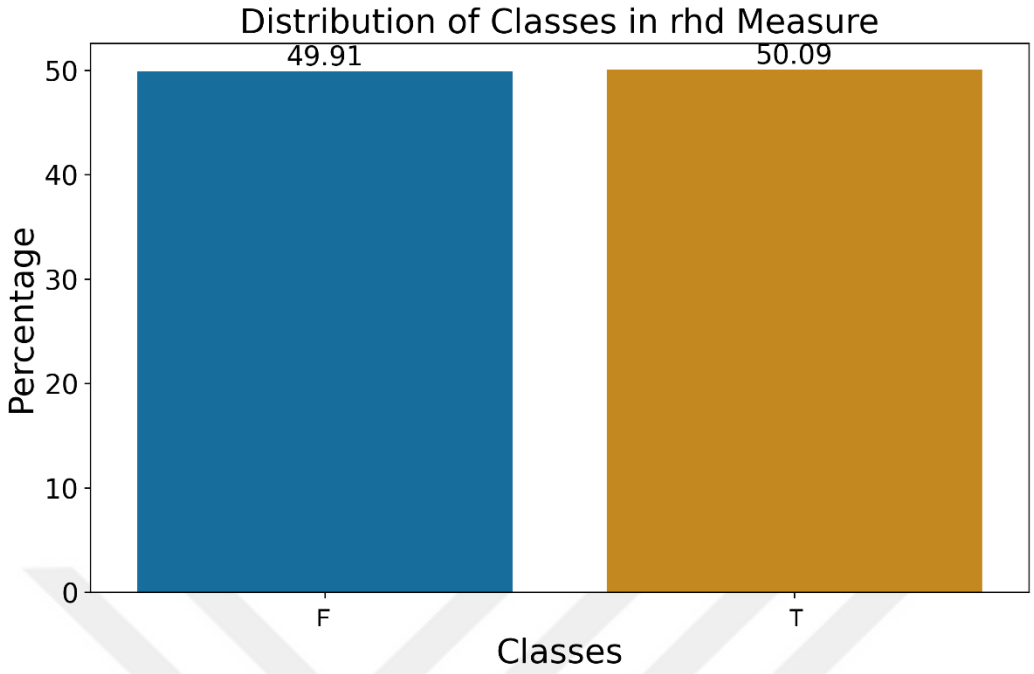

Figure 16. The class distribution of right-hand side heads case. The classes were distributed almost perfectly.

For making classifications, it is essential to have balanced class distribution. However, sometimes it is inevitable to have imbalanced classes due to the nature of the data. In this case, the researcher can consider this while reporting the results. For example, if there is an imbalance, it would be more informative to report a confusion matrix than a simple accuracy percentage. Also, there are some ways to deal with imbalanced data. If these methods make a difference in the classification task, they can be used too. In our case, we expect class imbalances since we can not control how many heads a sentence has.

When we looked into our ‘is it head’ classes, we saw a slight imbalance between non-head (F) and head (T) classes (Figure 15). Little more than 53% of the words are labeled as non-heads, and little less than 47% of the words are labeled as heads. On the other hand, in the right-hand side heads case, the classes' sizes are almost equal. (Figure 16). However, this is not the case when it comes to the left-hand side heads condition (Figure 17). The left-hand side case has the most significant class imbalance among the three dependency measures concerning

whether the word is head or not. We can introduce class weights to the SVC function to negate any possible negative effect of this imbalance. It would help to avoid predicting the frequent class more often than others just because it is more common in the data.

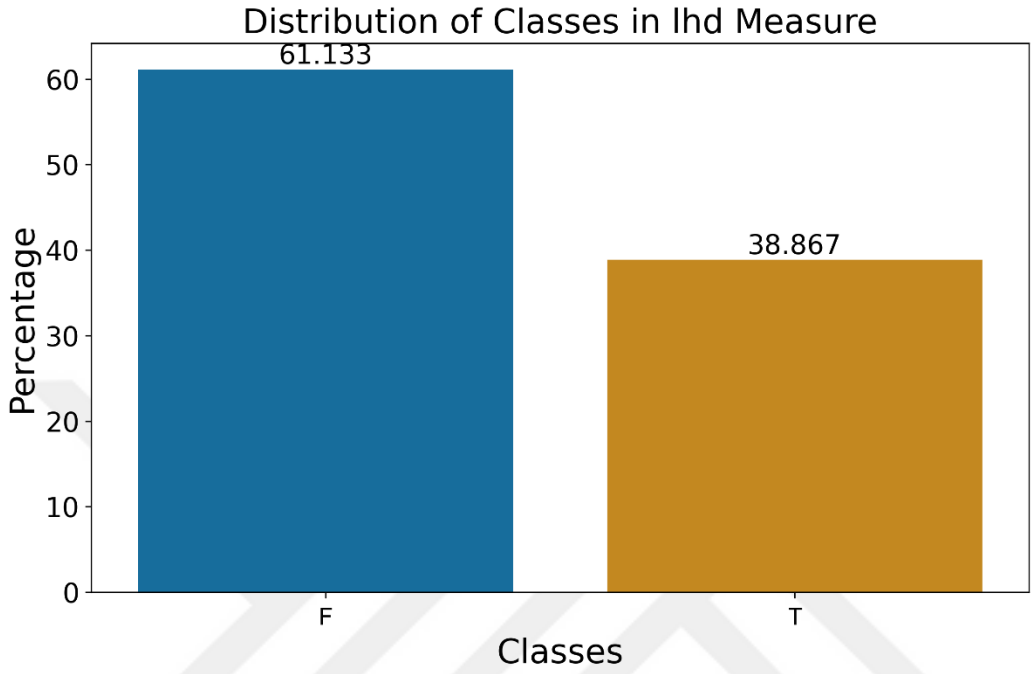

Figure 17. The class distribution of left-hand side heads. 61% of the words are in the non-left-head class.

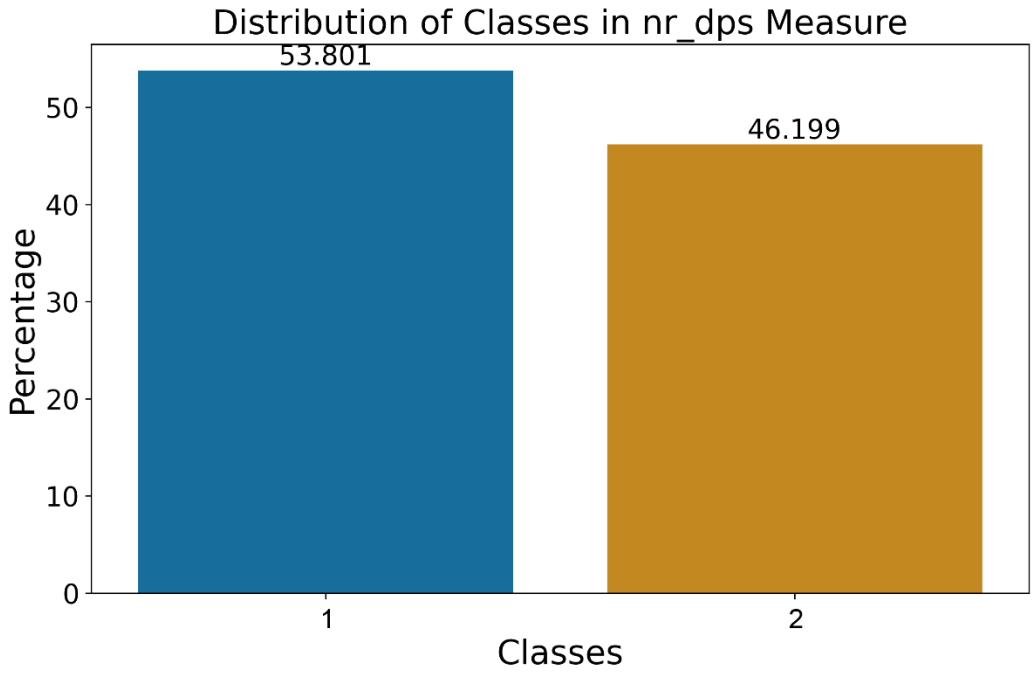

Figure 18. The class distribution of the number of dependencies case (nr_dps). The 1 class represents the words with one dependency relation. The 2 class represents the words with two or more dependency relations.

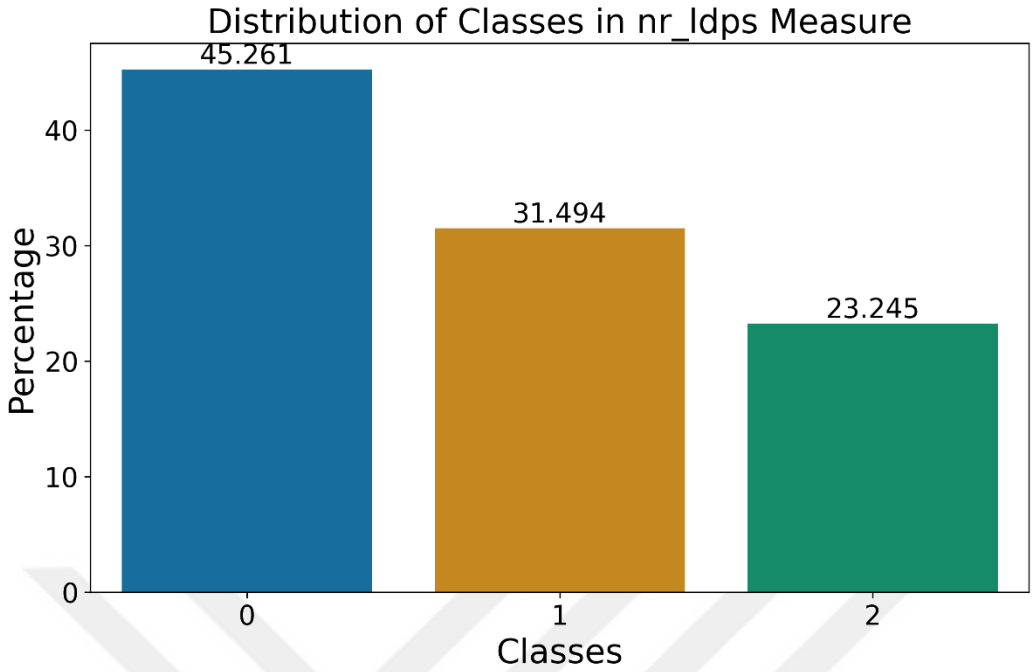

Figure 19. The class distribution of the number of left-hand side dependencies case (nr_ldps). The 0 class represents the words with zero dependency relation. The 1 class represents the words with one dependency relation. The 2 class represents the words with two or more dependency relations.

We had to make some adjustments for the dependency measures built around the number of dependency relations a word has. Mostly the words have one or two relations, but in some rare cases, some words have many relations. However, defining these rare cases as separate classes only creates more significant class imbalances. For example, suppose a class constitutes less than one percent of the overall data. This will create issues while training the probing classifier that might not be easily neutralized with methods like introducing class weights to computations. Therefore, we decided to group the classes of the dependency measures concerned with the number of dependency relations. We created two classes for the number of dependency relations measure. These are words with one dependency relation class and words with more than one dependency relation class. Since every word must have a relationship, there are no words with no relation class. As a result of this categorization, we basically turned the

number of dependency relations measure into a binary classification problem. Therefore we analyzed this measure with SVC. Figure 18 shows the class distribution for this measure.

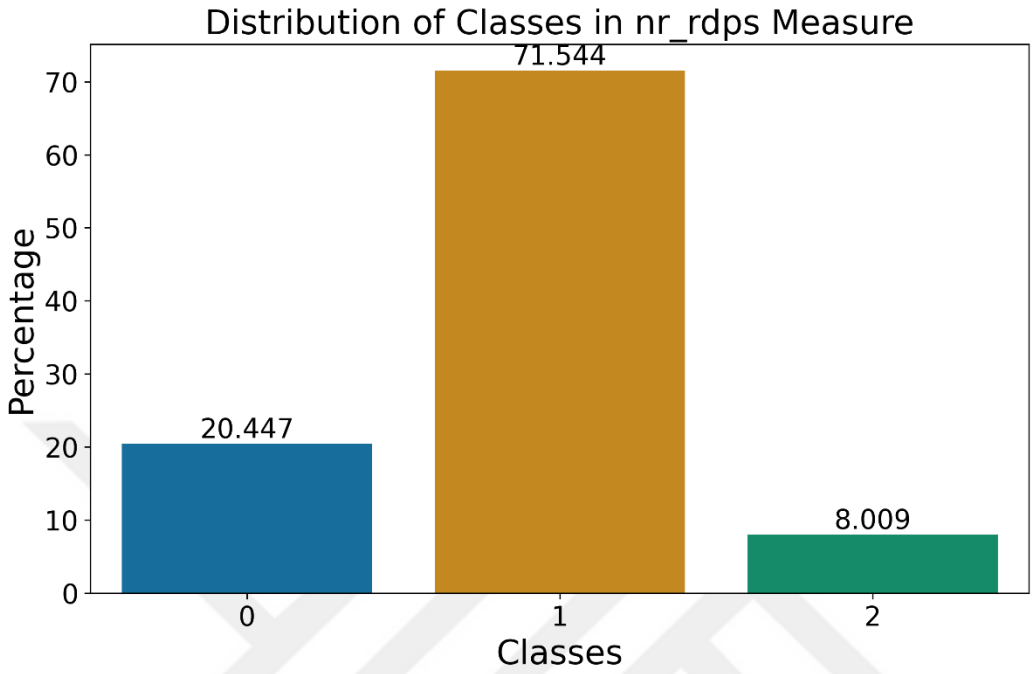

Figure 20. The class distribution of the number of right-hand side dependencies case (nr_rdps). The class representations are the same with the nr_ldps measure. It is clear that even after grouping, there is an imbalance in this measure.

We did a similar grouping for the number of left-hand and right-hand sides dependency measures. The only difference between these two measures has the possibility of zero relations. Even though every word must have a dependency relation, it is possible to have the relations only on the left-hand side and not on the right-hand side or vice versa. Therefore we grouped classes as the words that do not have any relation, the words that have only one relation, and the words that have more than one relation. This grouping still makes the number of left-hand side dependencies, and the number of right-hand side dependencies measures a multiclass classification problem. So, we used the multilayer perceptron as the probing classifier for these two cases. After the grouping of classes, of course, there was still some imbalance between classes. However, this imbalance was better than the imbalance before grouping. For the

number of left-hand side dependencies measure, the number of words that do not have any relationship is greater than the ones that have a relation (see Figure 19).

For the number of right-hand side dependency measure, it seems the majority of the words have one relation on the right side, and the words with no relationship and those with more than one relationship on the right are in the minority (see Figure 20). This is something we will take into consideration while interpreting the result.

6.2. Probing Classifier Analyses

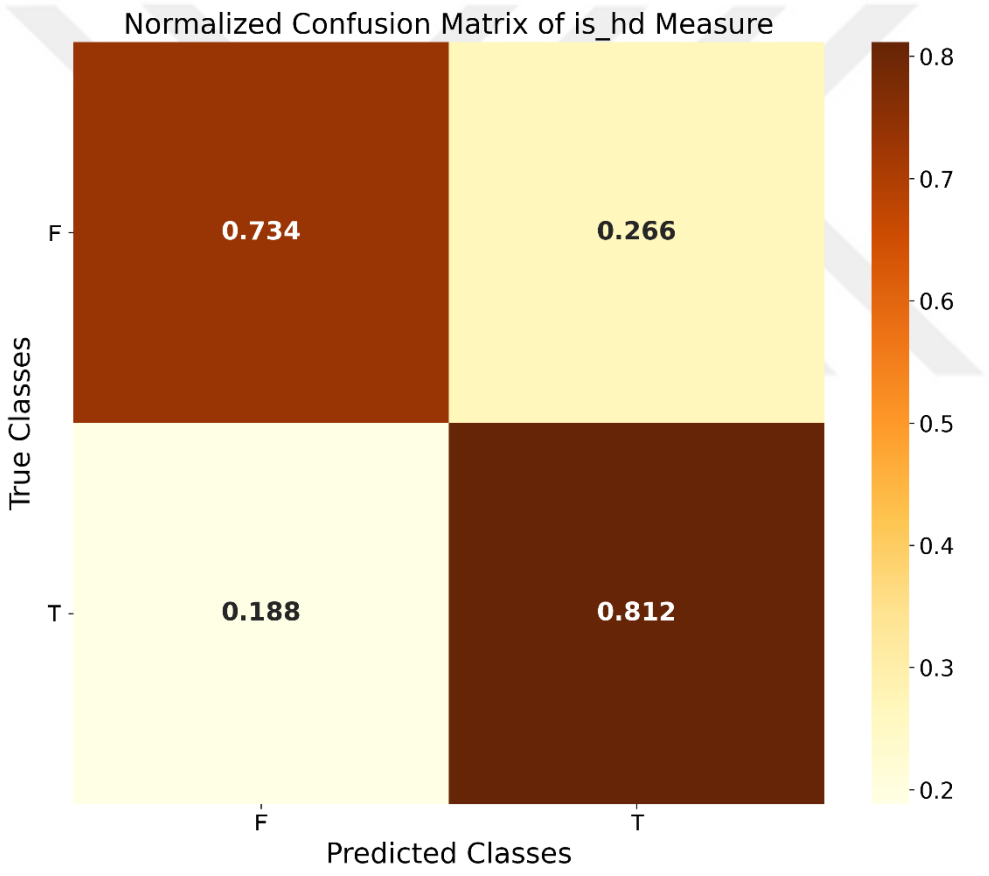

Figure 21. The confusion matrix for the 'is it head' measure. The normalization is done over true values (rows).

We used a randomized parameter search with the cross-validation (CV) method for each measure analyzed with SVC to find the best parameters. We determined the parameters for the

measures analyzed with MLP according to similar studies and their reports. For the ‘is it head’ measure, the randomized search method returned the best parameters as follows: C = 5.615, gamma = 0.036, and kernel = ‘radial basis function (RBF)’. With these parameters, the model has been trained again with unseen data using the CV method. Figure 21 shows the confusion matrix for the ‘is it head’ measure.

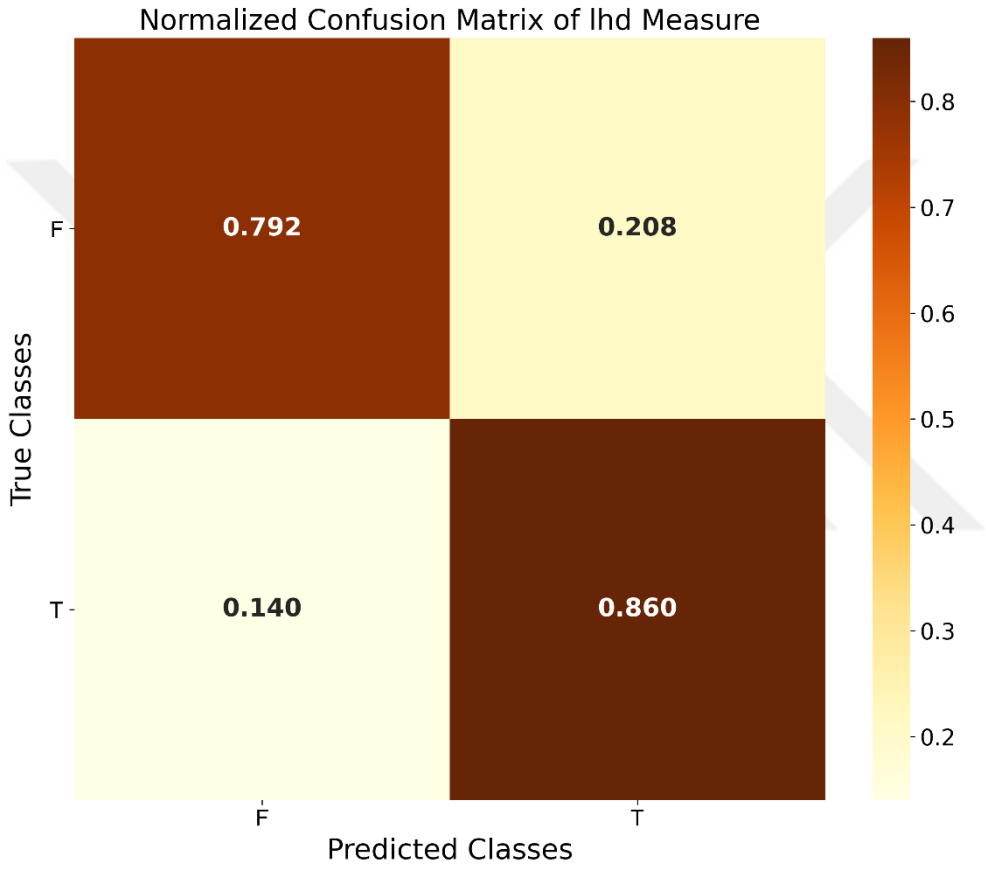

Figure 22. The confusion matrix for the left-hand side heads measure. The normalization is done over true values (rows).

The classifier correctly categorized little more than 70% of non-head words and 80% of head words. Almost 19% of the actual heads are classified falsely as non-head words, and 27% of the non-head words are classified as the head. The precision, recall, and f1 scores were also calculated for classifications. The precision score shows us, out of all positive class predictions,

how many of them actually belong to the positive class (which is the head class (a.k.a. class 1). The recall measures the fraction of our positive class that is categorized correctly. Finally, the f1 score gives us the harmonic mean of these two scores. The precision score for the 'is it head' measure is calculated as 0.72, and the recall returned as 0.81. The f1 score for this classification is 0.76. The result shows us that the SG model might actually be considering if a word is head or not.

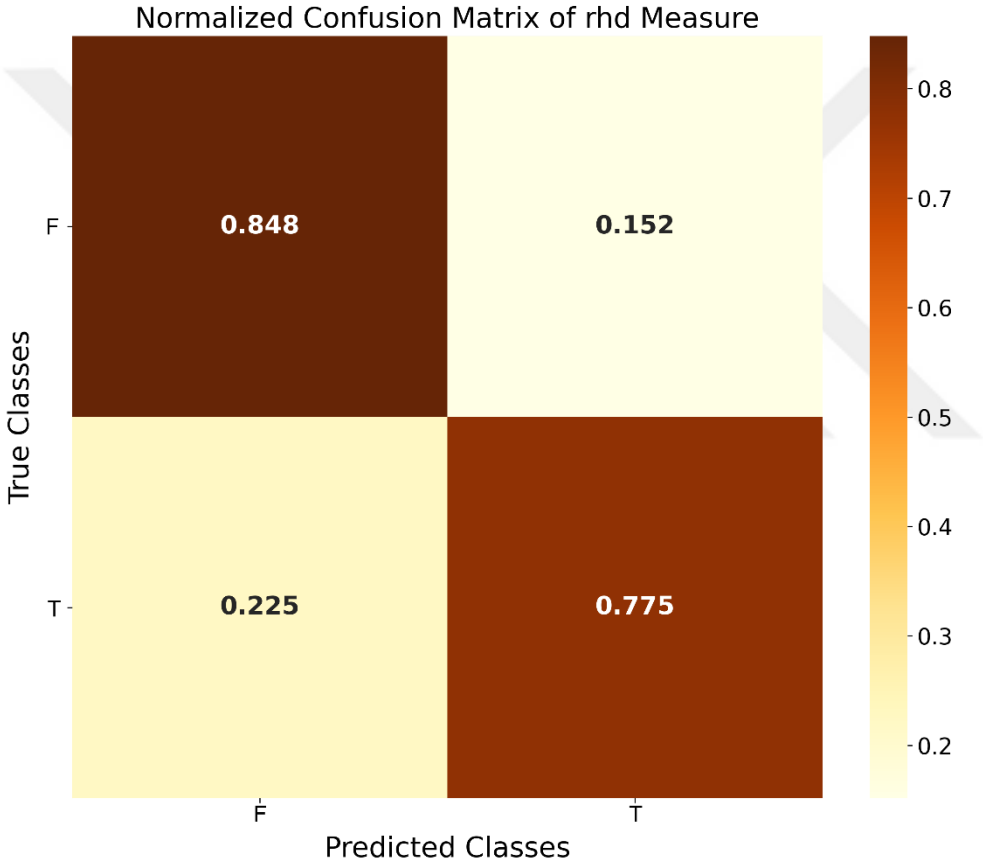

Figure 23. The confusion matrix for the right-hand side measure. The normalization is done over true values (rows).

For the left-hand side heads measure, the randomized parameter search returned the C value as 2.433, the gamma value as 0.078, and the kernel as RBF. With these parameters, the model was trained again using the CV process. Figure 22 shows the confusion matrix for the lhd

measure. The classifier correctly categorized 79% of non-left-hand side heads and 86% of left-hand side heads. For this classification task, the precision score is 0.72, the recall value is 0.85, and the f1 score is 0.78. These results show us, to some degree, that our classifier can detect the SG model's sensitivity if a head precedes its dependents or not.

The right-hand side heads measure analyzed in the same way. The selected parameters for the SVC classifier are as follows; C = 2.433, gamma = 0.078, and the kernel is RBF. After the training with selected parameters confusion matrix (see Figure 23) and other scores were calculated. The precision score returned as 0.83, the recall was 0.77, and the f1 score was 0.80. We were particularly curious about this measure. Since right-hand side heads are coming after their dependent, if the SG model processes its internal representations, this might mean the model has some anticipation about the upcoming dependency connection

So far, the current results suggest that the SG model might consider information about if a word is a head and what is the word's position regarding its dependent. Other than the word's role, the number of dependency relations might be another metric to look into since there are some examples in the literature. As mentioned above, in order to deal with the class imbalances, we did some grouping to our measures regarding the number of dependency relations. After this grouping, we had two classes for the number of dependency relations measure, which as a result of this, we analyzed with the SVC probe.

In the number of dependency relations condition, the training parameters for the SVC probe are determined as follows; C is 2.43, gamma is 0.078, and the kernel is RBF. The confusion matrix for this measure can be seen in Figure 24. To be clear, in this condition, class 1 represents the words with one dependency connection, and class 2 represents the words with two or more connections. Results show that 73% of the words with one connection and a little over %80 percent of the words with more than one connection are classified correctly. The precision and recall scores were calculated as 0.72 and 0.80, respectively. The f1 score returned as 0.76. This

result indicates that the SG model might be considering the number of dependency relations a word has.

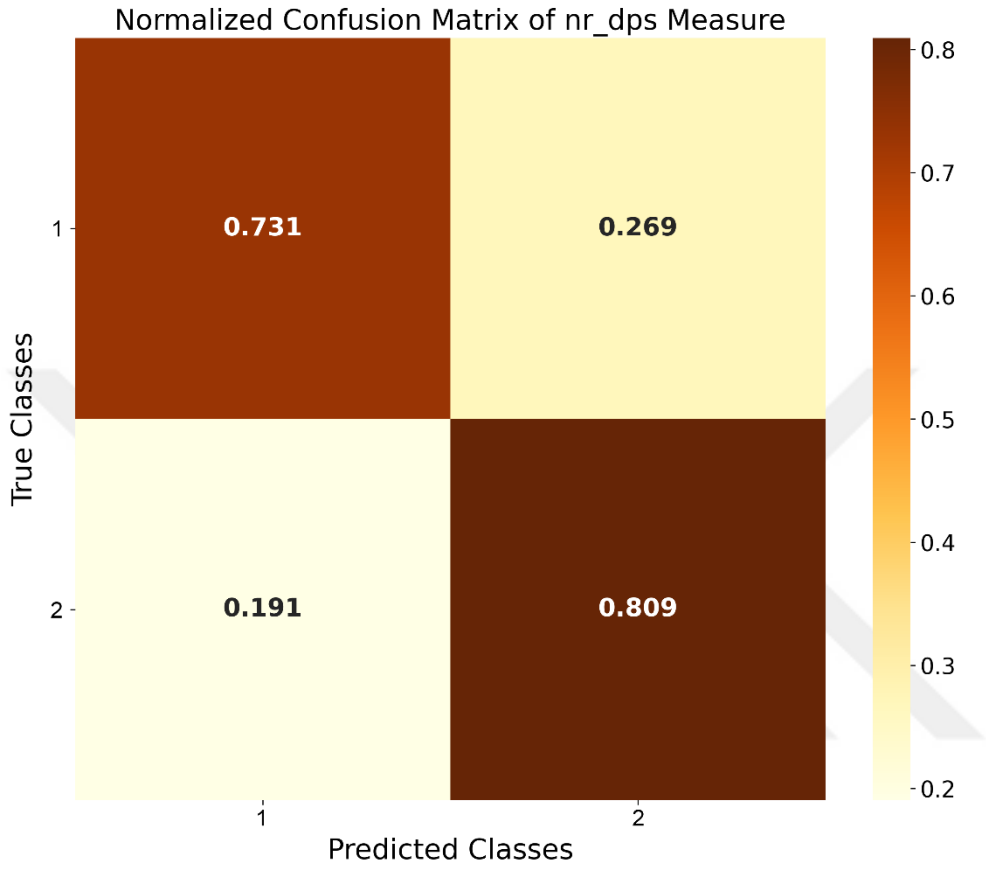

Figure 24. The confusion matrix for the number of dependency relations measure. The normalization is done over true values (rows).

Even after the class grouping, the left-hand side dependency relations measure was a multiclass classification problem. Therefore we analyzed this measure with the MLP probe. Class 0 represents the words with zero dependency relation, class 1 represents the words with only one dependency relation, and finally, class 2 represents the words with two or more dependency relations. 78% of the words that belong to class 0 are classified correctly. However, the number of correct classifications for classes 1 and 2 is much lower (see Figure 25). This result creates some doubts about how much information the SG model processes regarding the

number of left-hand side dependencies. The precision value for this classification returned as 0.69. The recall value was also 0.69, as well as the f1 score.

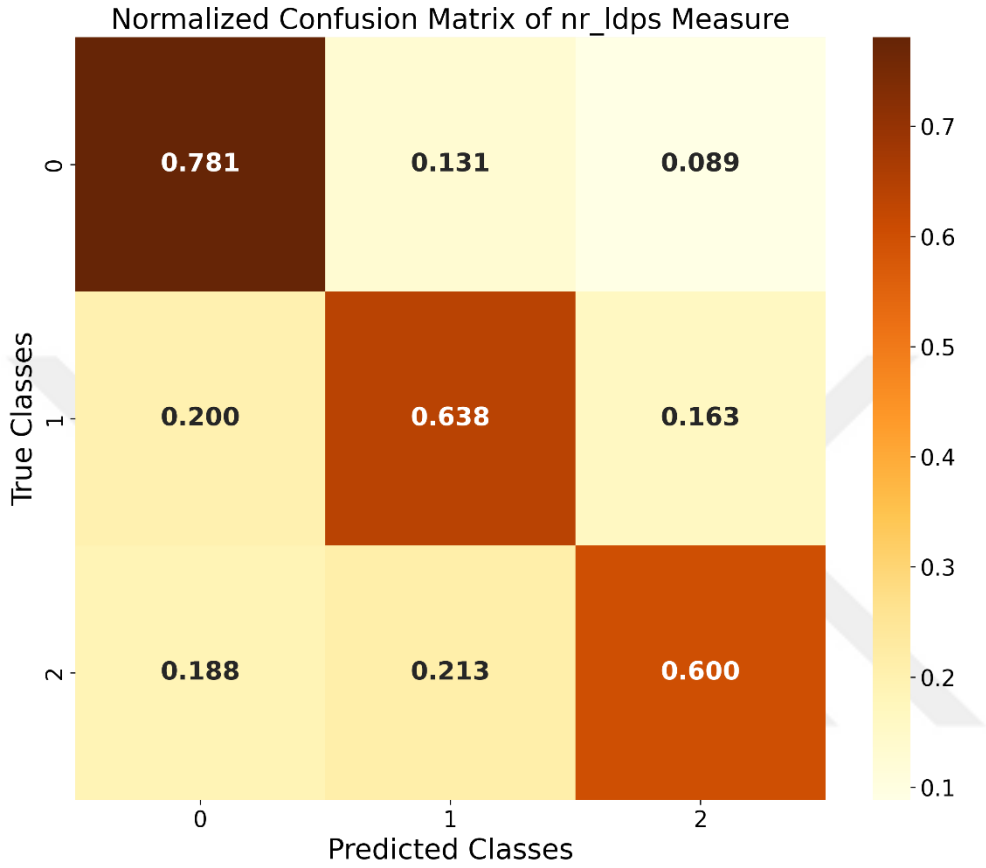

Figure 25. The confusion matrix for the number of left-hand side dependency relations measure. The normalization is done over true values (rows).

Lastly, we did our probing analysis for the number of right-hand side dependencies with the MLP probe. The classes are the same as the number of left dependency relations measure's classes. Although the probe correctly classified 84% of class 1, the classification performance for other classes was not satisfying. In fact, especially in class 0, there were quite a few misclassifications (see Figure 26). As a result, the precision and the recall scores for the number of left dependency relations measure are calculated as 0.71 and 0.72, respectively. From these values, the f1 score was computed as 0.72. Like the right-hand side heads measure, we were

curious about this measure. Since these relations follow the words being able to learn these connections would require some anticipation of incoming syntactic relations. However, similar to the left-hand side dependency relation, it is hard to interpret how much the SG model considers the right-hand side dependency relations.

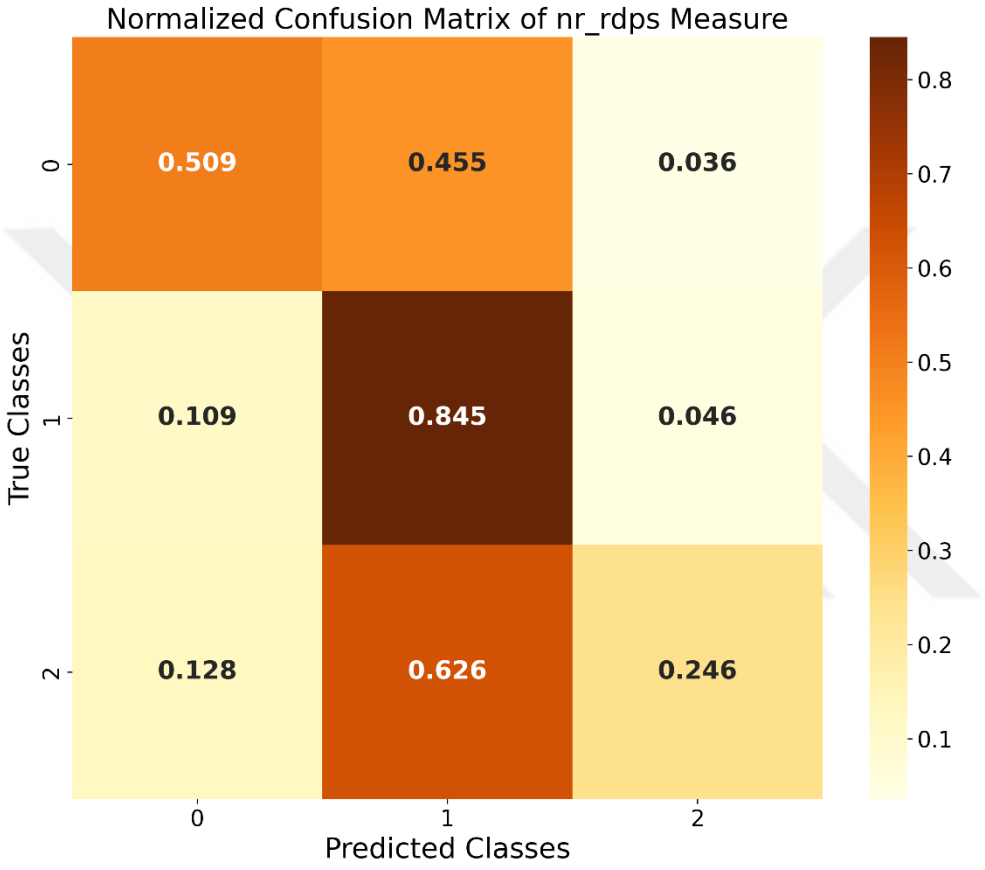

Figure 26. The confusion matrix for the number of the right-hand side dependency relations measure. The normalization is done over true values (rows).

Before moving to the discussion phase, it would be beneficial to summarize the results in a few sentences. The results for 'is it head', 'left-hand side heads', and 'right-hand side heads' measures suggest that the internal representations of the SG model might include syntactic information to a certain degree. On the other hand, the results for the 'number of dependencies', 'number of left-hand side dependencies', and 'number of right-hand side dependencies' are

slightly more mixed. For the number of dependencies measure, we have reasonable results that might indicate that this syntactic information is considered in the internal representations of the SG model. It also seems like the model's gestalt layer has some information about the number of left-hand and right-hand side dependency relations. Although, admittedly, it is hard to interpret. We will discuss these results in the next chapter to understand them better. We also summarized all the precision, recall, and f1 scores in Table 7, to make it easier to compare.

Table 7

Precision, Recall, and F1 Scores for all probing experiments

Measure name	Precision	Recall	F1 Score
is_hd	0.728	0.811	0.767
lhd	0.724	0.859	0.786
rhd	0.836	0.775	0.804
nr_dps	0.720	0.809	0.762
nr_ldps	0.693	0.693	0.692
nr_rdps	0.716	0.728	0.720

7. Discussion

The purpose of this thesis study was to answer the question of whether or not the Sentence Gestalt model implicitly processes syntactic information in its internal representations. The SG model is a cognitively motivated artificial neural network that can capture a few critical aspects of human language comprehension. It achieves this without having an in-built syntactic parsing mechanism or being provided with explicit syntactic information. Since we knew from linguistic studies that humans can implicitly learn syntactic information (Rebuschat & Williams, 2009; Kidd, 2012; Petersson & Hagoort, 2012), we were interested to see if the SG model would be able to capture this aspect of language comprehension too. To investigate this, we chose to work with dependency structures containing information about syntactic relations between words. We did probing experiments on the internal representations of the SG model to test our research question. Our results indicate that the SG model could take into account at least a few features of syntactic dependency measures. We will discuss our results in this chapter to further understand how they can be interpreted. Also, we will point out a few limitations and possible alternative explanations for our results.

7.1. Head and Non-Head Related Dependency Measures

First, we focused if a word's role in dependency relations (being head or not) is considered by the SG model or not. The result of our probing experiment indicates that the SG model might be processing this information. Additionally, we were interested in testing the SG model's sensitivity to a head's location concerning its dependant. Our research demonstrated that the SG model's representation contains information about if a head precedes or follows its dependent word. It was particularly interesting to observe that our probe can detect information in the internal representation of SG about heads that follow the dependent. This result might mean the SG model has some anticipation about incoming dependency connections. We mentioned in the introduction and background chapters that in psycholinguistics, some researchers theorize

that humans form some predictions about upcoming words during language comprehension (Van Berkum et al., 2005; Otten & Van Berkum, 2008; Pickering & van Gompel, 2010; Frank et al. 2015). Also, in various studies on the SG model, it has been recognized that the model form some expectations about incoming words based on the established semantic context (Rabovky et al., 2016, 2018; Lopopolo & Rabovsky, 2021). When all these are considered, the result of probing for right-hand side heads might be considered as further evidence for the SG model engaging in predictive processes. It seems these processes also occur at the syntactic dependency structures level.

On the other hand, we believe there might be some other explanations for the probing experiment results with measures that concern whether a word's role is head or not. The fact that a word is classified as a head might reflect that its Part-of-Speech (POS) is potentially encoded in internal representations of SG. For example, theoretically, it is more likely that verbs will be a head than other parts of the sentence (Nivre, 2005). Therefore, even though we created the 'is it head' and similar measures using syntactic dependency structures, the measure also can represent if a word is a verb or not. Hence, the probes might be finding representations of this verb information in the gestalt layer of the SG model. We believe this could be a thing to address in future works. The dependency parse we used to create our syntactic dependency measures also offers information on the part of the speech of words. It might be helpful to look into that information.

7.2. The Measures Related to the Number of Dependency Relations

Besides the word's role in dependency structure, we thought the number of dependency relations a word has could be important syntactic information and worth investigating. Lopopolo et al. (2021) used left-hand side dependency relations in their study as a feature of dependency structures and found that this measure explains some activity in specific brain regions. Our results suggest the SG model might be considering the number of dependency

relations. This could be interpreted as proof that the model implicitly considers syntactic information in its internal states. As we did with the ‘is it head’ measure, we came up with left-hand and right-hand side measures for the number of dependency relations. Our motivation was the same. We want to see if the model differs between already established and upcoming dependency relations. Even though their f1 scores are slightly less than the other measures, they can be considered satisfying. So we can state that the results indicate that the SG model might also consider these two syntactic information.

However, there might be some confoundings in the measures of left-hand and right-hand side dependency results. There is quite a bit of class imbalance in both of these measures. When we look at the confusion matrices, we can see that probes predicted the most common class better than the others. The situation can be observed better, especially in the right-hand side dependency relations case. In this measure, the most common class is class 1 (which is the class that has one right-hand side relationship). Little more than 84% of the words belonging to class 1 are predicted correctly. However, when we look at other classes, we see 45% of class 0 and 62% of class 2 also classified as class 1. Most likely, this situation arises from the fact that 71.5% of the words belong to class 1 in the right-hand side dependency relations measure. One solution to this problem might be resampling the classes to have an equal amount of words (Sun et al., 2009). However, we decided not to go through with this solution in our study. The main reason for this, we did not think there would be enough words left for efficient analysis after resampling. Having an adequate sample size is another crucial thing in machine learning studies. If resampling classes left a very small amount of words for classification, then it might not be wise to go through with it. If the sample size can be increased with more data, then reanalyzing left-hand and right-hand side dependency measures with resampling might lead to better results.

7.3. Future Directions

In this thesis study, we solely focused on layer-wise probings. However, this is not the only way to do probing experiments. For example, we mentioned in the introduction chapter that Arps et al. (2022) also focus on implicit syntax learning with a different artificial neural network model. Their study additionally includes a neuron-wise analysis to see if syntactic information is considered more in some neurons than others. This could be another future direction for studying this topic. Furthermore, even though syntactic information is present in the internal layers of the model, this might create more questions than answers about the SG model and syntax. For example, our study does not answer how much syntax-related information from the sentence gestalt layer affects the output layer and is used in predictions.

To sum up, this thesis contributes to the literature by providing evidence that the Sentence Gestalt model might be implicitly learning some syntactic dependency relations. However, there might be some alternative explanations for this result, and a more in-depth study might be required to reach better answers.

8. References

Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A. E., & Arshad, H. (2018). State-of-the-art in Artificial Neural Network Applications: A survey. *Heliyon*, *4*(11). https://doi.org/10.1016/j.heliyon.2018.e00938

Aggarwal, C. C. (2019a). An Introduction to Neural Networks. In *Neural networks and deep learning: A textbook* (pp. 1–52). chapter, Springer.

Aggarwal, C. C. (2019b). Machine Learning with Shallow Neural Networks. In *Neural networks and deep learning: A textbook* (pp. 53–104). essay, Springer.

Alain, G., & Bengio, Y. (2016). Understanding intermediate layers using linear classifier probes. *ArXiv Preprint ArXiv:1610.01644v3*.

Anderson, B. (2014). An Introduction to Neural Networks. In *Computational Neuroscience and Cognitive Modelling: A student's introduction to methods and procedures* (pp. 81–95). chapter, SAGE.

Ardesch, D. J., Scholtens, L. H., Li, L., Preuss, T. M., Rilling, J. K., & van den Heuvel, M. P. (2019). Evolutionary expansion of connectivity between Multimodal Association areas in the human brain compared with chimpanzees. *Proceedings of the National Academy of Sciences*, *116*(14), 7101–7106. https://doi.org/10.1073/pnas.1818512116

Arps, D., Samih, Y., Kallmeyer, L., & Sajjad, H. (2022). Probing for constituency structure in neural language models. *arXiv preprint*, arXiv: 2204.06201

Bejani, M. M., & Ghatee, M. (2021). A systematic review on overfitting control in shallow and deep neural networks. *Artificial Intelligence Review*, *54*(8), 6391–6438. https://doi.org/10.1007/s10462-021-09975-1

Belinkov, Y. (2022). Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, *48*(1), 207–219. https://doi.org/10.1162/coli_a_00422

Chauhan, V. K., Dahiya, K., & Sharma, A. (2018). Problem formulations and solvers in Linear SVM: A Review. *Artificial Intelligence Review*, *52*(2), 803–855. https://doi.org/10.1007/s10462-018-9614-6

Cherkassky, V., & Ma, Y. (2004). Practical selection of SVM parameters and noise estimation for SVM regression. *Neural Networks*, *17*(1), 113–126. https://doi.org/10.1016/s0893-6080(03)00169-2

Conneau, A., Kruszewski, G., Lample, G., Barrault, L., & Baroni, M. (2018). What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. https://doi.org/10.18653/v1/p18-1198

BNC Consortium (2007). The British National Corpus, version 3 (BNC XML Edition). https://doi.org/http://www.natcorp.ox.ac.uk

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, *20*(3), 273–297. https://doi.org/10.1007/bf00994018

Eliasmith, C. (2005). Moving beyond metaphors: Understanding the mind for what it is. *Cognition and the Brain*, 131–159.

Ferraresi, A., Bernardini, S., Baroni, M., & Zanchetta, E. (2008). Introducing and evaluating ukWaC, a very large web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google?* (pp. 47–54).

Fradkov, A. L. (2020). Early history of machine learning. *IFAC-PapersOnLine*, *53*(2), 1385–1390. https://doi.org/10.1016/j.ifacol.2020.12.1888

Frank, S. L., Otten, L. J., Galli, G., & Vigliocco, G. (2015). The ERP response to the amount of information conveyed by words in sentences. *Brain and Language*, *140*, 1–11. https://doi.org/10.1016/j.bandl.2014.10.006

Friedenberg, J., & Silverman, G. (2012). The Linguistic Approach: Language and Cognitive Science. In *Cognitive science: An introduction to the study of mind* (pp. 275–310). chapter, Sage.

Giulianelli, M., Harding, J., Mohnert, F., Hupkes, D., & Zuidema, W. (2018). Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. https://doi.org/10.18653/v1/w18-5426

Han, J., Kamber, M., & Pei, J. (2012). Classification: Advanced Methods. In *Data Mining* (3rd ed., pp. 393–442). essay, The Morgan Kaufmann.

Hewitt, J., & Liang, P. (2019). Designing and interpreting probes with control tasks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. https://doi.org/10.18653/v1/d19-1275

Hewitt, J., & Manning, C. D. (2019). A Structural Probe for Finding Syntax in Word Representations. In *2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Vol. 1, pp. 4129–4138). Minneapolis, Minnesota; Association for Computational Linguistics.

Hickok, G. (2009). The functional neuroanatomy of Language. *Physics of Life Reviews*, *6*(3), 121–143. https://doi.org/10.1016/j.plrev.2009.06.001

Hinton, G. E. (2011). Machine Learning for Neuroscience. *Neural Systems & Circuits*, *1*(12), 1–2. https://doi.org/10.1186/2042-1001-1-12

Jackendoff, R. (2019). Mental Representations for Language. In P. Hagoort (Ed.), *Human language: From genes and brains to behavior* (pp. 7–20). chapter, The MIT Press.

Kaan, E., & Swaab, T. Y. (2002). The brain circuitry of syntactic comprehension. *Trends in Cognitive Sciences*, *6*(8), 350–356. https://doi.org/10.1016/s1364-6613(02)01947-2

Kidd, E. (2012). Implicit statistical learning is directly associated with the acquisition of syntax. *Developmental Psychology*, *48*(1), 171–184. https://doi.org/10.1037/a0025405

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *ArXiv Preprint ArXiv:1412.6980*.

Kotsiantis, S. (2007). Supervised Machine Learning: A Review of Classification Techniques. *Informatica*, *31*, 249–268.

Liu, Y. (2006). *Create Stable Neural Networks by Cross-Validation*. International Joint Conference on Neural Networks. Vancouver, Canada.

Lopopolo, A., & Rabovsky, M. (2021). Predicting the N400 ERP component using the sentence gestalt model trained on a large scale corpus. *BioRxiv*. https://doi.org/10.1101/2021.05.12.443787

Lopopolo, A., van den Bosch, A., Petersson, K.-M., & Willems, R. M. (2021). Distinguishing syntactic operations in the brain: Dependency and phrase-structure parsing. *Neurobiology of Language*, *2*(1), 152–175. https://doi.org/10.1162/nol_a_00029

Mahesh, B. (2020). Machine Learning Algorithms - A Review. *International Journal of Science and Research*, *9*, 381–386. https://doi.org/10.21275/ART20203995

Mallot, H. A. (2015). Artificial Neural Networks. In *Computational neuroscience: A first course* (pp. 83–112). chapter, Springer.

Matchin, W., & Hickok, G. (2019). The cortical organization of Syntax. *Cerebral Cortex*, *30*(3), 1481–1498. https://doi.org/10.1093/cercor/bhz180

McClelland, J. L., St. John, M., & Taraban, R. (1989). Sentence comprehension: A parallel distributed processing approach. *Language and Cognitive Processes*, *4*(3-4). https://doi.org/10.1080/01690968908406371

Mitchell, T., Buchanan, B., Dejong, G., Dietterich, T., Rosenbloom, P., & Waibel, A. (1990). Machine Learning. *Annu. Rev. Comput. Sci*, *4*(4), 17–33.

Nabiyev, N., & Malekzadeh, S. (2021). Anomalous sound localization estimation. *Preprint*. https://doi.org/10.13140/RG.2.2.25949.95201

Nivre, J. (2005). Dependency Grammar and Dependency Parsing (MSI Report No. 05133). Växjö, Sweden: Växjö University.

Otten, M., & Van Berkum, J. J. (2008). Discourse-based word anticipation during language processing: Prediction or priming? *Discourse Processes*, *45*(6), 464–496. https://doi.org/10.1080/01638530802356463

Oxford University Press (a). (n.d.) Language, 2a. *Oxford English dictionary*. Retrieved from https://www.oed.com/view/Entry/105582?rskey=4XGJsd&result=1#eid

Oxford University Press (b). (n.d.) Corpus, 3b. *Oxford English dictionary*. Retrieved from https://www.oed.com/view/Entry/41873?redirectedFrom=corpus#eid

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Petersson, K. M., & Hagoort, P. (2012). The neurobiology of syntax: Beyond string sets. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *367*(1598), 1971–1983. https://doi.org/10.1098/rstb.2012.0101

Pickering, M. J., & van Gompel, R. P. G. (2010). Syntactic parsing. In M. J. Traxler & M. Gernsbacher (Eds.), *Handbook of psycholinguistics* (pp. 455–503). chapter, Elsevier.

Pylkkänen, L. (2019). The neural basis of combinatory syntax and semantics. *Science*, *366*(6461), 62–66. https://doi.org/10.1126/science.aax0050

Pylkkänen, L., & McElree, B. (2010). The syntax-semantics interface: On-line composition of sentence meaning. In M. J. Traxler & M. Gernsbacher (Eds.), *Handbook of psycholinguistics* (pp. 539–579). chapter, Elsevier.

Rabovsky, M., Hansen, S. S., & Mcclelland, J. L. (2016). N400 amplitudes reflect change in a probabilistic representation of meaning: Evidence from a connectionist model. In *Annual Meeting of the Cognitive Science Society* (pp. 2045–2050). Philadelphia, PA.

Rabovsky, M., Hansen, S. S., & McClelland, J. L. (2018). Modelling the N400 brain potential as change in a probabilistic representation of meaning. *Nature Human Behaviour*, *2*(9), 693–705. https://doi.org/10.1038/s41562-018-0406-4

Radu, M. D., Costea, I. M., & Stan, V. A. (2020). 12th International Conference on Electronics, *Computers and Artificial Intelligence (ECAI)* (pp. 1–4). IEEE.

Rebuschat, P., Williams, J. N., Taatgen, N. A. (Ed.), & Van Rijn, H. (Ed.) (2009). *Implicit learning of word order.*. 425-430. Paper presented at Proceedings of the 31st Annual Conference of the Cognitive Science Society.

Sayeed, A., Shkadzko, P., & Demberg, V. (2018). *Rollenwechsel-English: a large-scale semantic role corpus*. European Language Resources Association. http://dx.doi.org/10.22028/D291-30972

Sharma, S., Sharma, S., & Athaiya, A. (2020). Activaiton Function in Neural Networks. *International Journal of Engineering Applied Sciences and Technology*, *4*(12), 310–316.

Skansi, S. (2018). Feedforward Neural Networks. In *Introduction to deep learning: From logical calculus to artificial intelligence* (pp. 79–105). chapter, Springer.

Sun, Y., Wong, A. K., & Kamel, M. S. (2009). Classification of Imbalanced Data: A Review. *International Journal of Pattern Recognition and Artificial Intelligence*, *23*(04), 687–719. https://doi.org/10.1142/s0218001409007326

Szandała, T. (2020). Review and comparison of commonly used activation functions for deep neural networks. *Bio-Inspired Neurocomputing*, 203–224. https://doi.org/10.1007/978-981-15-5495-7_11

St. John, M. F., & McClelland, J. L. (1990). Learning and applying contextual constraints in sentence comprehension. *Artificial Intelligence*, *46*(1-2), 217–257.

Traxler, M. J. (2012). *Introduction to psycholinguistics: Understanding language science*. Wiley-Blackwell.

Vallender, E. J., Mekel-Bobrov, N., & Lahn, B. T. (2008). Genetic basis of human brain evolution. *Trends in Neurosciences*, *31*(12), 637–644. https://doi.org/10.1016/j.tins.2008.08.010

Van Berkum, J. J., Brown, C. M., Zwitserlood, P., Kooijman, V., & Hagoort, P. (2005). Anticipating upcoming words in discourse: Evidence from Erps and reading times.

Journal of Experimental Psychology: Learning, Memory, and Cognition, 31(3), 443–467. https://doi.org/10.1037/0278-7393.31.3.443

Vapnik, V. N. (2000). Methods of Pattern Recognition. In *The nature of statistical learning theory* (2nd ed., pp. 123–180). essay, Springer.

Van Rossum, G., & Drake, F. L. (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace.

Wang, H., Ma, C., & Zhou, L. (2009). A brief review of machine learning and its Application. *2009 International Conference on Information Engineering and Computer Science*. https://doi.org/10.1109/iciecs.2009.5362936