On machine learning in linguistics: how artifical neural networks compare to human language processing

Scriptie 2020

Joey Moes

Introduction

Artificial Neural Networks have made great strides in a lot of different fields the last decades. From beating grand masters in Chess and Go, to self-driving cars and predicting all sorts of statistics such as mortality rate or length of stay in hospitals (Rajkomar et al. 2018). The field of Natural Language processing has not been left unexplored. Deep learning models in the field of natural language processing (NLP) are now able to successfully translate, transcribe and produce texts of a high quality. Since language was thought to be a species-specific ability of mankind for so long, new and old questions arise in the field of NLP. The main question that I will be trying to answer in this paper is: Do artificial neural networks and humans process language in the same way? If they do, this will change the way we think about the nature of language processing for good. Language would stop being something species-specific. Other agents might make use of language in the same manner if they have a similar neural network configuration. In this paper I will use many different studies from linguistics and computational linguistics to compare the linguistic capabilities of humans and of deep neural networks (DNNs). After a short introduction, the concept of syntax is explained. I will try to illustrate why syntax exists and what kind of evidence there is for it. To compare the syntactical abilities of humans and DNNs I will first explain how humans use syntax. After a short explanation of DNNs I will illustrate if and how DNNs use syntax. In the conclusion, I will be able to make a full comparison.

Human language processing

To answer our question of whether artificial neural networks use the same syntactic structure as humans, we will first need to elaborate on how humans process a language and use syntax. A child learns its first language(s) without the language(s) being taught explicitly. Parents do not need to teach a child a language, children will learn the language without any explicit lessons. However, the parents of a child play an important role in the acquisition of language by talking to their child. The child also needs some sort of interaction with the language and language-speakers. Without interaction, like by only listening to the radio or the television, the child will not learn a language. However, if two children that have not learned to speak a language are put together, they will naturally make their own language (Yang et al. 2017). Thus, language is an inherent skill all humans share, not a construct created by society. In order to process language, we have some inherent capability but we need experience and interaction with a language to be able to process it fully. In the same way that children gain a vocabulary, they also gain an understanding of the syntax of the language that they are exposed to. However syntax is something that children have knowledge of naturally. While children get better at reading syntactic cues (Li et al., 2020), they already use syntax to understand the most common language structures from the moment they can speak simple phrases. A formal definition of syntax is that it is the grammatical rule structure that underlies a language. Whether syntax is also gained through experience or whether there are inherent rules that underlie it, has been a subject of debate for a long time. Noam Chomsky has been the most influential person in this debate. Noam Chomsky (2005, from Yang et al., 2017) argues that, given the relatively brief history of homo sapiens, and their species-specific computational capacity for language. Language and its evolution is most likely built on a foundation of other cognitive abilities that humans have. Language acquisition is, according to Chomsky (2005, from Yang et al., 2017), heavily reliant on three factors:

1- **Universal Grammar (UG)**: the acquisition of language is heavily dependent on our genetic endowment. There are certain rules that have evolved within our brain, this helps us in our learning and understanding of language.
2- **Experience** : by experiencing language first hand and interacting with it, humans gain knowledge of the language. This factor is the reason of why there are so many different languages.
3- **Third factors**: Language acquisition is made possible by the different cognitive abilities that we have had before we learned to use language. Such abilities include hypothesis formation and data analysis. Not only this, but the efficiency of our brain and external factors also play a role.

Universal grammar

The first factor on universal grammar might need some more explaining. To answer the question of how universal grammar would have come into existence, UG is likely to be the result of evolution and natural selection (Bolhuis, 2017). Where just like in every species that has undergone evolution, the species that mutated to have UG was more likely to develop languages and thus had a better chance of survival. The argument against this, as Bolhuis mentions, might be that major biological systems would need a very long time to evolve (Christiansen et al., 2008). Critics like Christiansen et al. argue that since language is relatively young, this could not be the result of evolution, which needs millennia before a muta-

tion can cause an entire species to adapt and evolve. Critics of the evolution argument argue that language and UG has been "shaped to fit the human brain, and not vice versa" (Christiansen et al., 2008). What they mean is that UG has been produced by mankind for mankind, to make language easier to process, and thus is a construct of society, not something that has evolved over years of evolution. However, the evolution of UG may have been a much more simple than a complex process. Because of this simplicity, it would not have needed a lot of years to evolve, a simple mutation without evolution could have been enough to cause the hierarchical structure of language (Bolhuis 2017).

Merge This mutation is thought to be a rewiring of the brain which allowed the operation of 'merge', which is not a language specific theory and may also be used with mathematics or logic. "Merge is a (dyadic) operation that takes two syntactic objects, call them X and Y, and constructs from them a single new syntactic object, call it Z. X,Y can be building blocks that are drawn from the lexicon or previously constructed objects. Put simply, Merge (X,Y) just forms the set containing X and Y. Neither X nor Y is modified in the course of the operation Merge."(Everaert et al., 2015) So, merge is a way to make phrases have a hierarchical structure by adding objects, which can be words or phrases, to a new object structure without removing any part of the added objects. Since merge is not a language specific concept, this further solidifies the idea of language acquisition being made possible not only by experience but also by UG and the third factors, which are cognitive domains/abilities that humans have had before we learned how to use language, which play an important role in our use of language.

Generative grammar Merge is a simple concept that makes the language that we use today possible. However, it does not encompass the entirety of UG. Merge in itself does not explain why children have an innate ability to learn languages quickly and thoroughly, while adults often will not be able to achieve as much when they decide to learn a new language. This might be because when the adults learn a new language they try to use the rules they have learned from their first language, instead of using their innate linguistic knowledge. So, what is this innate linguistic knowledge or UG? UG is a direct result of the generative grammar linguistic theory. This theory regards language as having innate grammatical structure as we have mentioned before. With this structure, true and meaningful sentences can be created instead of a sentence with words but no inherent meaning. A phrase like "the cat" has meaning while a phrase like "cat the" has no meaning. These structures can be laid out with parse trees. A parse tree for the first sentence can be seen in figure 1.

Here, the 'NP' stands for 'Noun Phrase', a noun phrase consists of a determiner and a noun, as mentioned before it also needs to be in that order, which is exactly what the parse tree shows. In this manner, a parse tree can be created

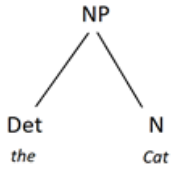

Figure 1: A parse tree of the noun phrase "The cat"

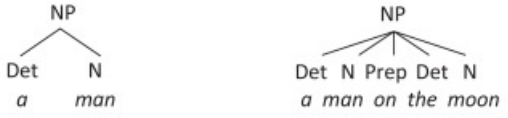

Figure 2: NP = Noun phrase, Det = Determiner, N = Noun and Prep = Preposition

for every sentence. There need to be as many 'leaves' in the parse tree as there are words in the sentence. The leaves then connect via branches and converge to a single point which is 'S', which stand for 'sentence', if it would be a complete sentence, unlike our example. Generative grammar is also recursive, this stems directly from the property of merge. (Everaert et al. 2015). I will use the figures and explanation from Everaert's paper to demonstrate. As we have seen, phrases can be structured by phrase structure rules. For the parse trees show in figure 2, we can make two rules that describe the grammar of the sentences:
1- NP $\rightarrow$ Det N
2- NP $\rightarrow$ Det N Prep Det N

As you can see, a sentence like "A man on the moon" can theoretically go on forever by just adding prepositions another determiner and a noun. For example: "A man on the moon with the friend with a flag by the ship". We would then need to add another rule for our language which would be:
3- NP $\rightarrow$ Det N Prep Det N Prep Det N Prep Det N Prep Det N

Recursion Making rules based on each sentence is not an efficient way of describing the grammar of a language. Thus, we could also generalize this rule by making a simpler grammar that is recursive. Such as:
1- NP $\rightarrow$ Det N (PP) (Noun phrases consist of a determiner and a noun, but may also be followed up by a prepositional phrase)
2- NP $\rightarrow$ Prep NP (prepositional phrases consist of a preposition and a noun phrase)
As Everaert et al. mentions, while this recursiveness means that language has the potential to be infinite, it "should not be incorrectly equated with real-time production or parsing of actual utterances". This means that while generative grammar allows us to make an infinitely long sentence, this, does not occur in the speech of humans. You may have noticed that grammaticality does not always entail a semantically correct

sentence. In the sense that a sentence, that is grammatical, does not always have meaning.

Semantics in syntax A property of syntax is that semantics and syntax are very often intertwined. With words that are called negative polarity items (Everaert et al., 2015) such as the word "anybody", there needs to be a negative element that gives them meaning. You can't say "Anybody liked my song". To make the sentence structurally correct it would need to be "My song was not liked by anybody", elsewise the word 'anybody' would need to be changed. This is further evidence that language is never just linear and requires an actual structural hierarchy for it to have meaning and make sense.

UG through DNNs A very common way to study linguistic structures is by simulation. We will go into detail about deep neural networks (DNNs) in a later section, however, it is important to mention that through simulation we can find arguments that support the UG theory. Lepori and colleagues (2020) found that when comparing constituency DNN models and dependency DNN models the former model performed a lot better. This is another argument that supports UG since the constituency model is based on hierarchical structure, such as parse trees, like UG. The dependency model is based on a linear representation of the data, this does not coincide with the view that grammar has an inherent hierarchical phrase structure.

Wh-licensors A different property of syntax is that in some sentences, a gap where a word could be is created. For example: "I know who _ did that ", the follow up to this would be: "I know **that** you did that". The 'who' in the first word is called a wh-licensor, having a word like this in a sentence would make it ungrammatical if the object/subject that the wh-word refers to would also be mentioned. For example: "*I know who you did that". We will come across this phenomenon, along with a multitude of other phenomena, in an upcoming section where we will be discussing syntax in DNNs. The main takeaway from this section is that by analyzing language and syntax, we can clearly see that languages are "structures, not strings" (Everaert et al., 2015).

Syntax in the brain

To find concrete and hard evidence of syntax, researchers have tried to use brain-imaging techniques to figure out where and how syntax is used within the brain. These techniques have been very successful in finding the location of the brain regions that are involved in syntactical processing (Grodzinsky & Friederici, 2006). Other researchers have researched the effect of brain lesions on the ability to process language (Caplan, 2006, Hagoort, 2005). The processing of syntactical information has been generally localized to two distinct areas, the inferiorfrontal gyrus and the superiortemporal cortex (Friederici et al., 2000). However, while the question of where syntactic processing is located within the brain is interesting, for linguists, it does not answer the most important questions. The question of how syntactic processing works in the brain is more important to this paper as well.

A way to research this is to show the test-subject a grammatical sentence and an ungrammatical sentence and see how the brain reacts differently to this. The problem that comes with this type of research, is that you can never be certain what the brain really reacts to. To make a sentence ungrammatical, the word order needs to change, the brain might just be reacting to an unexpected change in word order. One way to research the reaction to ungrammatical sentences is the degree of expectedness. We will mention this **surprisal** effect again in our analysis of syntax in DNNs. In short, the rate of surprisal is determined by the chance that a certain word will appear after a some other word, after the word "dog" we would sooner expect the word "leash" than the word "soup". To truly see if the brain reacts to grammaticality or just uncommon sentences, a distinction needs to be made between rule- and probability related processes.

In research done by Pulvermüller & Assadollahi (2007) they found direct neurophysiological evidence that supports a system in the brain that uses rules for words and morphemes and not just statistical processing. Using magnetoencephalography (MEG) they report a syntactic mismatch negativity (sMMN) that "distinguished syntactic violations from common grammatical strings, but not uncommon from common grammatical strings". This is evidence that the brain reacts to grammaticality and not just uncommon strings.

Recursion in the brain In a previous subsection we have talked about the language property that is recursion. Recursion requires a sort of stack data structure or a push-down algorithm, where items that are stored first are the last ones that are retrieved. To see if there is any neurophysiological basis for recursion within the human brain, Braitenberg et al. (1992, from Pulvermuller, 2010) researched memory circuits. Memory circuits can be activated serially and will gradually lose activity. This would make the last activated circuit have the most activity and the first activated circuit the least. To be able to handle recursion, the brain only needs to have a read out mechanism that prioritizes the highest activity in the circuit when accessing the circuits. This way, the read out mechanism will recursively go back on the activated circuits in a last-in-first-out manner. This stack reading could be done in parallel with multiple stacks at the same time (Joshi & Schabes, 1997).

All in all, there is enough evidence to know where the process of syntax takes place and how it would be able to work within the brain. It is very difficult to use any other kinds of more concrete research methods like a single cell scan. Right now, it is impossible to find out exactly which neuron responds to which input, whether that is a word, a syntactical property or something entirely different. The question that we will try to answer in the next section will be: do DNNs also use syntactical structure in their language processing.

Artificial Neural Networks

In this section, I will explain how NNs work and how they are used in language processing. For this, I will use the explanation provided by Linzen et al. (2020). Neural networks are mathematical functions, these functions receive an input, which is a sequence of real numbers, and output a different sequence of real numbers. They do so by having a large collection of "computation units" which you can compare to neurons, each of these neurons calculates "the weighted average of its input". The function that is computed by each unit/neuron is $\sigma(w_1x_1 + \ldots + wnxn)$, where the w stands for the weight and the x stands for the input. While each calculation in each neuron is very simple and easy, the complexity emerges when the neurons are ordered into layers and one layer's output can act as the next layer's input, this is how much more complex functions are able to be computed. The NNs weights are not set by the designer of the network, but rather inferred by learning from examples. To use a NN, first, training needs to take place. Here, the weights will change according to inputs (x_i) and their expected values(y_i). The weights will start with a random value but will change with each iteration. The weights will change when the output $\hat{y}$ does not match the expected value y, they will be changed in a way that the difference between $\hat{y}$ and y will be smaller, which is also known as "gradient descent". This process will keep going until no weights are changed for the entire process. After that, these weights can be used on a test-dataset, the model will be able to make predictions using the input from the testdataset and the calculated weights from training.

Vectors As mentioned before, DNNs are mathematical and thus only compute real numbers. To be able to interact with words in language processing, all words need to be translated into numbers, or as a vector, which is a sequence of numbers. The vector of the word "Horse" might be (2, 8, 9, 3.4), in this way, words that are used in a similar context can have a vector that has values which are close. In this way, "zebra" or "saddle" might have a vector (2, 8, 9.5, 4), while a word like "computer" might be (-4, 3.7, -6.3, 1). These values are not inputted by the designer of the ANN, rather these vectors are learned using gradient descent. In this manner, words can be encoded. As there are thousands of words in a language, the vectors can become very long. In order to encode sentences or bigger language structures, we will need something more complex.

Recurrent Neural Networks This is where recurrent neural networks come in (RNN). RNNs process a sentence from left to right. In this process, a vector (ht) is maintained, this vector is " a hidden state which represents the first t words of the sentence" (Linzen et al., 2020). This acts as a kind of for-loop and a bottleneck, where ht+1 is calculated from ht and the next word in the sentence, this way the RNN does not have access to its previous states. To further improve upon RNNs capabilities, gated networks are used. An example of this is Long Short-Term Memory networks (LSTM). First introduced by Hochreiter et al. in 1997, LSTMs serve as a solution to previous systems that had problems with "back-flow errors". These back-flow errors are caused by common errors being "blown up" or vanishing, when using back-propagation, this can "lead to oscillating weights" or having to learn to bridge long time lags, which may not work and is very time consuming. A LSTM is a RNN that uses cells, input gates, output gates and a forget gate. By deciding to forget about blown up or vanished errors, no more back-flow errors can occur. LSTMs are the most common type of RNN used today.

When RNNs are used in NLP, they are mostly used for three different purposes. It can be used as a classifier, where the network has to label a sequence 'acceptable' or 'unacceptable', mostly in terms of grammaticality. The network can also be used as a language model. In such a model, the network is asked to assign probabilities to each next word. In a sentence such as "the kids love to pet the . . . ", such a model will assign a higher probability to the last word being "cat" than a word such as "table". The last setting a network may be put in is a "sequence to sequence" or "seq2seq" setting. In this setting, the network is expected to generate its own output sequence, based on the input sequence it has received.
RNNs have had surprisingly good results on a lot of different NLP tasks, currently, achieving human performance (Young et al. 2018). However, their success is the direct result from their representations of the data, which are hard to interpret. A lot of research is being done to try to get a glimpse into what these RNNs data representations look like. Since syntax plays a vital role in human language processing, it seems more than natural to wonder if the success of RNNs is (partly) based on their ability to encode something that is similar to syntax.

Grammatical structure in RNNs

So, the question is, do these RNNs encode a "hierarchical context-free phrase structure" instead of some superficial use of word order that is based on a probability for each word in each sentence. To research this, several studies have been conducted. Each of these studies uses a different strategy to find out how the RNN is representing its input.

Agreement Some studies measure the state of the NN as it is processing specific inputs. These inputs have been chosen or made specifically to try and measure the processes of the NN, for example: "using number agreement in subject-verb dependencies" (Linzen et al., 2016). In sentences like "the cat is eating", the verb needs to agree with the subject in terms of number. In a sentence like "The cat are eating" this clearly is not the case, the subject or the verb needs to be changed in this sentence to be grammatically correct. In their research, Linzen and colleagues found that the NN made incorrect number predictions in less than a percent of

the dependencies, but this might be because the verb and subject are so close together. Linzen and colleagues have experimented with making larger sentences as inputs. A sentence like "The cat that had gotten fatter every day since the lovers adopted it are eating" might be harder for a NN to encode or 'understand', since "the lovers" might be seen as the subject of the verb "are", the noun "lovers" is therefore an *attractor*. In such cases where they used up to four attractors, the NN had an error rate of 84%, which is less than chance. Linzen and colleagues' conclusion was that LSTMs are just sequence models and "they do not have built-in hierarchical representations".

Semantics in RNNs In research done by Bernardy et al. (2017), where they experimented with LSTMs and different DNN structures such as GRUs and CNNs, they found similar results to the research done by Linzen. However, they also found that DNNs "require large vocabularies to form substantive lexical embeddings in order to learn structural patterns". According to Bernardy and Lappin, this suggests that DNNs would be more efficient in learning syntactic patterns through more extensive lexical embeddings, that have syntactic as well as semantic cues. This is a very different strategy than the one used before where strings are made up of simple words to make a specific sentence structure. However, if a DNN relies on semantic information, while getting better results, this is a sign that DNNs do not encode a syntactical structure like humans do, since grammatical sentences can exist without being semantically correct.

Gulordova et al. (2018) did more research to find out if RNNs truly depend on these semantic cues to make good predictions in the long distance agreement-task. In their research they used standard corpus extracted examples as well as semantically incorrect sentences. Quoting them, they were inspired by Chomsky's argument that "grammaticalness cannot be identified with meaningfulness". As such, they used meaningless sentences such as: "The colorless green ideas I ate with the chair sleep furiously". In such a sentence "ideas" and "sleep" belong together, and the RNN should classify them as such. Their results have been very convincing, while the RNN's predictions were not perfect, they also were not lagging far behind the human performance. In Italian, the RNN performed especially well, getting more than 92% correct in the meaningless sentences while also getting 93,3% correct in the standard corpus extracted examples. English was the hardest language for the RNN, however, they explain that this might be due to a lot of English sentences being slightly ambiguous. In sentences such as "if you have any questions or need/needs", the target could be a noun as well as a verb. The results did not differ greatly from that of human achievements, though this comparison was only made with the Italian results.

The question that arises now, is whether this is enough evidence to say that DNNs use syntactical structure in their language processing methods. The fact that DNNs performed well in a lot of agreement dependencies would certainly suggest so. However, more research is needed to conclude this with a better degree of certainty.

Surprisal So far, we have only seen research conducted in agreement dependencies, to truly get an idea of DNNs syntactical capabilities it is important to look into research methods that have included other syntactical properties as well. In research done by Wilcox et al. (2018) they studied, using LSTMs, filler-gap dependencies. Being able to show that DNNs are able to implement filler-gap dependencies would bring us a step closer to being able to say that DNNs encode syntactical structure to process language, like humans do. Filler gap dependencies use wh-licensors, we have mentioned these shortly in our section on syntax in humans. Wh-licensors are words like "who", "what", "why", "which" etc. and gaps. These gaps are places within a sentence where a word could be, but whether we should fill the word in the gap depends on whether a wh-licensor was used. In every sentence where a wh-licensor is used, there should be a gap, represented as an underscore: '_'. The examples used in the research done by Wilcox and colleagues are:

a) I know that the lion devoured a gazelle at sunrise.
b) * I know what the lion devoured a gazelle at sunrise.
c) * I know that the lion devoured _ at sunrise.
d) I know what the lion devoured _ at sunrise.

In these examples, we can see that in each sentence where a wh-licensor is used, a gap is needed instead of "a gazelle" to make the sentence grammatically correct. In the sentences without a wh-licensor, using a gap instead of "a gazelle" results in an ungrammatical sentence.

The way the LSTMs are tested is by using a measurement called surprisal, a surprisal value can be assigned to every word and sentence by an RNN. The value of the surprisal can tell us whether the word or sentence was unexpected for the RNN. In their paper, they researched whether the surprisal, that is caused by an unusual sentence construction, like a gap, would have a smaller surprisal value in the presence of a licensor. "If the models learn that syntactic gaps require licensing, then sentences with licensors should exhibit lower surprisal than minimally different pairs that lack a proper licensor" (Wilcox et al. 2018). They measure the surprisal at the word that immediately follows the (filled) gap and over the entire sequence of the sentence after the gap to the end of the embedded clause. So, the surprisal is not measured at "a gazelle" but rather at "at sunrise". Their results show that the LSTMs surprisal was higher when the sentence was ungrammatical. Not only with the previous type of sentences, but also with sentences where the object, subject or indirect object was extracted, such as:

a. I know who _ showed the presentation to the visitors yesterday (object extraction)
b. I know what the businessman showed _ to the visitors

yesterday (subject extraction)
c. I know who the businessman showed the presentation to _ yesterday (indirect object extraction)

They also used island constraints, which is another syntactic configuration the LSTMs could be tested on. An 'island' is a clause or a structure where a word, or most commonly a noun phrase, cannot be removed from. For example, in the following sentences it would be ungrammatical for a gap to appear inside a sentence that has doubly nested clauses with a wh-licensor: "I know what Alex said whether your friend devoured _ at the party" If in this sentence the word "whether" was ejected or replaced by "that", the sentence would not be ungrammatical. There are a lot more constraints that they tested the LSTM on. On all these constraints the LSTM performed well, at least as well as human performance. With the exception of the 'subject island' constraint. Under this constraint, a prepositional phrase, following a noun phrase, can only contain a gap if the subject of the sentence is not the noun phrase, these examples are taken from the paper by Wilcox et al. (2018)

a) I know what _ fetched a high price at auction.
b) *I know who the painting that depicted _ fetched a high price at auction.
c) *I know who the painting which depicted _ fetched a high price at auction.
d) *I know who the painting by _ fetched a high price at auction.

Their results show that LSTMs can learn to represent filler-gap dependencies and their constraints. Since they did not use some inductive bias, they argue that this can also be seen as evidence that an inductive bias is not necessary for language processing. Thus, this would represent a threat to UG being necessary for language processing. They do warn however, that the amount of data that they have used for training, is a lot larger than the amount of data that can serve as input for a child learner.

Testing other syntactical phenomena In a follow up study done by mostly the same researchers (Futrell et al, 2019), they researched the effects of *subordinate clauses* and *garden path effects*. A subordinate clause is a sentence that announces a following clause, like: "When I brushed my hair, …" A garden path sentence is a sentence which is grammatically correct but, due to the structure of the sentence, lures the reader into an incorrect interpretation. A sentence such as "The old man the boat" can be seen as grammatically or semantically incorrect when really it is not. Having a comma in place after "old" would make the sentence more readable. To study this, they used 4 different models which all had different results.

- JRNN (Jozeficz et al., 2016), this model is based on the LSTM architecture and has a large data-size of about 800 million tokens.
- GRNN (Gulordova et al, 2018), this model is also based on the LSTM architecture but has a smaller data-size of about 90 million tokens from Wikipedia.
- RNNG (Dyer et al. 2016), the architecture that this model is based on is called RNN Grammar, and has a training data size of about 1 million tokens
- TinyLSTM, this model is a version of LSTM that has a small training data-size of only 1 million tokens.

The differences between these models can tell us if the amount of training data plays a big role in whether or not the RNN is able to correctly encode syntactical structure. In their research they found that all of these models were able to correctly deal with subordination and garden path sentences. However, the TinyLSTM model was not able to deal with "more fine-grained phenomena". The JRNN could not deal with the more fine grained phenomena of subordination, such as the no-matrix penalty effect. The RNNG was not able to deal with the more fine grained phenomena of some of the garden path sentences, such as verb-transitivity. Overall, the outcome of this study is fairly positive, many different RNNs are able to deal with subordinate clauses and garden path sentences, where the less common and more complex phenomena require a lot more data for the RNN to be able to handle correctly. The RNNG had relatively good results while only being trained with 1 million tokens. This is a direct result of the pre-programmed syntactical structure within the RNN (Dyer et al., 2017). That RNNs perform a lot better on tasks that require syntactical structure when the model has a built in structure is further reinforced by McCoy et al. (2020). In their research, they found that models where they implemented a parse-tree structure, showed the most clear bias towards a hierarchical structure.

Inner workings of a DNNs

Thus far, we have seen that DNNs respond well to most of the grammatical phenomena they are tested on. These phenomena are used to indicate whether or not a DNN has a syntactical representation. Since, the DNN arguably would not be able to correctly process these phenomena if it did not. The phenomena that are used are the same phenomena that are used to indicate syntax in human language processing, as we have seen in our first section on syntax of humans. A different way that we have discussed, to study syntax of humans, was to analyze the inner brain processes of humans. To truly and fully compare the syntactical capabilities of humans and AI/DNNs we will have to analyze the inner workings of DNNs as well. Immediately, we can assume that analyzing the inner workings of a DNN, several of the problems we have encountered with the human analysis, will be a lot easier to deal with while analyzing the DNNs. A major improvement is that we would not need to have any expensive equipment to peer into the inner workings of a DNN. However, since the information is not encoded as an easy to analyze structure,

such as parse trees, but rather in vectors that can contain a hundred real numbers, there is still plenty of hardship in analyzing the inner workings of a DNN. In order for us to analyze anything, the vectors would need to be translated into a readable format. Which has proven to be very difficult. However, there are some studies that have made an attempt to do this.

Regarding this, the first study we will discuss is by Shi et al. (2016). To find out if the encoder learns syntactical information about the source sentence and what kind of information is learned, they used two different methods. Firstly they created syntactical labels of the source sentence, such as: NP, VP etc. They tried to predict the syntactic labels with logistic regression. For sentence level labels they used the learned sentence encoding vectors, and for the word-level labels, they used the word-by-word hidden vectors. The second step is to extract a full parse tree from the source sentence from the encoded vectors. This way, a structured manner to analyze the information that is encoded can be represented. The results of this study show promise, a lot of syntactical information is encoded. In their research they find that different types of syntactical information is encoded in different layers of the DNN. However some syntactical information is still missing. This causes the most common error in their research, which is called "sense confusion". Missing information has a lot of effect on the rest of the parse tree. As can be seen in the example they shared which is shown in figure 3. In this example the word "beyond" is predicted as an RB instead of an IN, which causes a missing prepositional phrase (PP), and thus causes sense confusion.

In research done by Conneau et al. (2018) they used similar probing tasks to uncover information from the DNN. There were several tasks that tested specific things. Firstly, a bigram shift task, this task "tests whether an encoder is sensitive to legal word orders". The model needs to be able to tell the difference between intact sentences and a sentence where a random pair of words is switched, for example: "What *you are* doing out there?". The second task is the tree depth task, in this task the encoder needs to be able to retrieve the maximum depth of a parse tree of the sentence. The last task about syntax was the top constituent task. In this task, the encoder needs to be able to formulate the first couple of constituents and say what sort of constituent it is, such as NP, VP etc. A lot of the encoders did well on all of these tasks.

Giullianelli et al. (2018) revealed that the 'diagnostic classifiers', which are trained on number agreement from the internal state of the model, "provide a detailed understanding of how, when and where this information is represented". In their research they find that it can also be shown where wrong information is generated and how that influences the model into making a mistake. They used the extracted information to help the LSTM during the processing of a difficult sentence and found that it drastically increases the accuracy of the model.

Saved information usage This does point out a problem that these methods face. While the methods discussed can show that a DNN has syntactical information stored, that does not necessarily mean that it uses that information. For example, in the previous study they had to manually feed the stored information back into the model to help it get better results. Funnily, this sound like a very human mistake, storing information without using it.

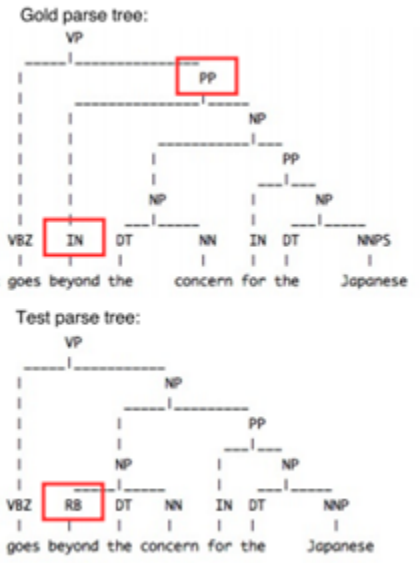

Figure 3: The upper parse tree shows the expected result and the lower parse tree shows the tree that the DNN outputted.

While not only reinforcing the previously mentioned findings of the research done by Gulordova et al. (2018), Lakretz and colleagues (2019) found that when tracking individual neurons or computation units, after some training on raw corpus data, the units began tracking very specific linguistic information. The number information is managed by a pair of neurons, so whether the phrase is single or plural. These neurons in turn are "partially controlled by other units independently shown to track syntactic structure". So, each neuron keeps track of some aspect of the syntactical information of the sentence. In turn these neurons influence the information in a different neuron. This (sub)network is supported by other (sub)networks that track non-syntactical information such as linear distance. While the model still had some trouble with the tracking of plurality of a sentence with embedded phrases, this research has the most convincing results. Not only showing that the model is able to deal with grammatical tasks, but also being able to point out exactly which neurons do what and the role that they play in the entire network.

Conclusion

All in all, the research in NLP that has been done using DNNs in the last couple of years has changed the consensus on the topic completely. A few years back in 2017, the conclusion of Linzen's and Bernardy's first research was that DNNs do not

encode syntactical structure and rather rely on large amounts of data to fuel their conclusions in a statistical manner. I think it is safe to say that most of the experts of the topic agreed with this conclusion even before Linzen's and Bernardy's research. However, in the 3 years after, an entirely new conclusion was being drawn from other sources. Gulordova has proven that RNNs can handle long-distance agreement tasks, even with meaningless sentences. To reinforce the conclusion that RNNs encode syntactical information within the model, a lot of different syntax tasks were tested. Using the surprisal calculation method, Wilcox (2018) and Futrell (2019) had very positive results. Though Futrell's research does point out that larger training-datasizes will still perform better on the more complicated syntax constraints. However, models that have a built in hierarchical structure such as a parse-tree structure, were the most human-like in their processing of language (Futrell et al., 2019, McCoy et al., 2020). Finally, Lakretz and colleagues were able to show definitively that the network encodes syntactical information and uses it as well. Their model, however, did encounter some problems with embedded phrases, since this model was still based on a linear structure.

As research and models develop, it seems clear that DNNs are able to process language with fairly minimal mistakes. The more data the DNNs are trained on, the less mistakes they make. The only way to make a DNN make relatively few mistakes without making them train on billions of tokens, since humans have less training data as well, is to implement a hierarchical bias into the model. This further reinforces the argument of UG and the three factors that Chomsky (2005, from Yang et al., 2017) mentioned are necessary for language acquisition/processing. So, artificial neural networks can process language in a similar hierarchical manner as humans, however, just as humans have a bias towards hierarchical language processing in the form of UG, evidence suggests that the model needs to have a bias towards such a hierarchical based structure, like an implemented parse-tree structure, to have a chance of correctly processing language in this manner.

References

Bolhuis, J. J. (2017). Making sense of language in the light of evolution. *Mind & Language*, *32*(5), 591–596.

Caplan, D. (2006). Aphasic deficits in syntactic processing. *Cortex*, *42*(6), 797–804.

Christiansen, M. H., & Chater, N. (2008). Language as shaped by the brain. *Behavioral and brain sciences*, *31*(5), 489–509.

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. *arXiv preprint arXiv:1805.01070*.

Dyer, C., Kuncoro, A., Ballesteros, M., & Smith, N. A. (2016). Recurrent neural network grammars. *arXiv preprint arXiv:1602.07776*.

Friederici, A. D., Wang, Y., Herrmann, C. S., Maess, B., & Oertel, U. (2000). Localization of early syntactic processes in frontal and temporal cortical areas: a magnetoencephalographic study. *Human brain mapping*, *11*(1), 1–11.

Futrell, R., Wilcox, E., Morita, T., Qian, P., Ballesteros, M., & Levy, R. (2019). Neural language models as psycholinguistic subjects: Representations of syntactic state. *arXiv preprint arXiv:1903.03260*.

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. *arXiv preprint arXiv:1808.08079*.

Grodzinsky, Y., & Friederici, A. D. (2006). Neuroimaging of syntax and syntactic processing. *Current opinion in neurobiology*, *16*(2), 240–246.

Gulordava, K., Bojanowski, P., Grave, E., Linzen, T., & Baroni, M. (2018). Colorless green recurrent networks dream hierarchically. *arXiv preprint arXiv:1803.11138*.

Hagoort, P. (2005). On broca, brain, and binding: a new framework. *Trends in cognitive sciences*, *9*(9), 416–423.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Joshi, A. K., & Schabes, Y. (1997). Tree-adjoining grammars. In *Handbook of formal languages* (pp. 69–123). Springer.

Jozefowicz, R., Vinyals, O., Schuster, M., Shazeer, N., & Wu, Y. (2016). Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*.

Lakretz, Y., Kruszewski, G., Desbordes, T., Hupkes, D., Dehaene, S., & Baroni, M. (2019). The emergence of number and syntax units in lstm language models. *arXiv preprint arXiv:1903.07435*.

Lepori, M. A., Linzen, T., & McCoy, R. T. (2020). Representations of syntax [mask] useful: Effects of constituency and dependency structure in recursive lstms. *arXiv preprint arXiv:2005.00019*.

Li, J., & Zhou, P. (2020). Children's interpretation of ambiguous wh-adjuncts in mandarin chinese..

Linzen, T., & Baroni, M. (2020). Syntactic structure from deep learning. *arXiv preprint arXiv:2004.10827*.

Linzen, T., Dupoux, E., & Goldberg, Y. (2016). Assessing the ability of lstms to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, *4*, 521–535.

McCoy, R. T., Frank, R., & Linzen, T. (2020). Does syntax need to grow on trees? sources of hierarchical inductive bias in sequence-to-sequence networks. *Transactions of the Association for Computational Linguistics*, *8*, 125–140.

Pulvermüller, F. (2010). Brain embodiment of syntax and grammar: Discrete combinatorial mechanisms spelt out in neuronal circuits. *Brain and language*, *112*(3), 167–179.

Pulvermüller, F., & Assadollahi, R. (2007). Grammar or serial order?: Discrete combinatorial brain mechanisms reflected by the syntactic mismatch negativity. *Journal of Cognitive Neuroscience*, *19*(6), 971–980.

Rajkomar, A., Oren, E., Chen, K., Dai, A. M., Hajaj, N., Hardt, M., ... others (2018). Scalable and accurate deep learning with electronic health records. *NPJ Digital Medicine*, *1*(1), 18.

Shi, X., Padhi, I., & Knight, K. (2016). Does string-based neural mt learn source syntax? In *Proceedings of the 2016 conference on empirical methods in natural language processing* (pp. 1526–1534).

Wilcox, E., Levy, R., Morita, T., & Futrell, R. (2018). What do rnn language models learn about filler-gap dependencies? *arXiv preprint arXiv:1809.00042*.

Yang, C., Crain, S., Berwick, R. C., Chomsky, N., & Bolhuis, J. J. (2017). The growth of language: Universal grammar, experience, and principles of computation. *Neuroscience & Biobehavioral Reviews*, *81*, 103–119.

Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, *13*(3), 55–75.