Syntactic properties of skip-thought vectors

Bart Broere
ANR: 429124

Master Thesis series nr. None

Thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Communication and Information Sciences,
Master Track Data Science: Business and Governance,
at the School of Humanities and Digital Sciences
of Tilburg University

Thesis committee:

Dr. Grzegorz Chrupała
Dr. Afra Alishahi

Tilburg University
School of Humanities
Department of Communication and Information Sciences
Tilburg center for Cognition and Communication (TiCC)
Tilburg, The Netherlands
January 2018

Syntactic properties of skip-thought vectors

Bart Broere

Abstract

Sentence representations are a powerful method of extracting fixed-width vectors from sentences, and are useful in domains ranging from machine translation to image retrieval. Evaluating the qualities of sentence representations is often done by employing the representations in standardised semantic prediction tasks. Syntactic properties are to a lesser extent subject to evaluation.

Here we evaluate the representations generated by one particular model; the skip-thoughts model (Kiros et al, 2015). This study addresses the lack of syntactic evaluation by employing intermediate hidden states of the skip-thoughts model in artificial prediction tasks. The experiments in this study show that information about syntax is present in skip-thought vectors. Logistic regression trained on the hidden states of the skip-thoughts model is capable of classifying grammatical categories. However, logistic regression optimised on word embeddings with context could still outperform this.

Despite the lack of word isolation in the hidden states of the skip-thoughts model, predicting grammatical categories has proven to be a promising method of evaluating syntactic properties present in sentence representations.

1 Introduction

Sentence representations are a type of feature vectors that are increasingly useful in a wide range of applications (Hill et al, 2016; Le, Mikolov, 2014), ranging from paraphrase detection to image retrieval. One such model, with a unique approach is skip-thoughts (Kiros et al, 2015).

The skip-thoughts model (Kiros et al, 2015) maps sentences that share semantic and syntactic properties to similar vectors. This is achieved by training the model to predict the surrounding sentences. It is an example of an encoder-decoder model (Goodfellow et al, 2016). The encoder part results in a hidden state h_i^t (a fixed-length vector). This hidden state forms the input to the decoder, which in training skip-thoughts maximises the log-likelihood of the surrounding sentences (s_{i-1} and s_{i+1}).

In the case of the skip-thoughts model, the decoder is only required for training the model. Afterwards, it is discarded and the encoder can be used to generate sentence representations. Skip-thoughts therefore is an unsupervised model; it does not require labeled data. It merely needs a continuous body of text. The setup of the skip-thoughts model is inspired by the way word embeddings are optimised (Kiros et al, 2015; Mikolov et al, 2013). The context of the sentences is assumed to be predictive of the content of the sentence in skip-thoughts, as the context of words is assumed to be predictive of the content of words in word embeddings (Mikolov et al, 2013).

Most experiments with sentence representations evaluate the semantic qualities of the generated sentence representations, fewer investigate the syntactic properties (Kiros et al, 2015; Gan et al, 2017).

The main question of this thesis that follows from this problem is: *To what extent are syntactic properties encoded in skip-thought vectors?* This main research question can be divided in several subques-

tions.

- *Can logistic regression trained on skip-thoughts hidden states correctly classify grammatical function of words?*
- *Does the aforementioned classification outperform a baseline of a classifier trained on word embeddings?*
- *What are the differences in (mis)classification between skip-thoughts hidden states and word embeddings?*

This exploratory research aims to address the lack of syntactic evaluation of models of sentence representation. The syntactic properties of skip-thought vectors are estimated by predicting part-of-speech (POS) and dependency tags. A pretrained model available in SpaCy (2017a) has been used to obtain the tags, which will be the target labels in the new artificial prediction task. This results in two multiclass classification problems; one for the POS tags, and one for the dependency tags. One-versus-rest logistic regression models are optimised on these problems.

To allow for comparison of the results, a similar classifier has been trained with the word embeddings of the skip-thoughts model. This classifier is expected to model syntactic effects present at word-level. The hidden states classifier is expected to model sentence-level syntactic effects, on top of the word-level. Where the skip-thoughts model encodes sentence structure, the classifier trained on hidden states should outperform the classifier trained on word embeddings.

The experiments indeed show that the hidden states of the skip-thoughts model offer an advantage in predicting tags. For POS tags, accuracy is only slightly better. For the more grammatically fine-grained dependency tags however, the classification improves significantly, compared to the single word embeddings classifier.

A qualitative comparison of the error between the models for predicting POS tags shows that this improvement is mainly caused by ambiguous words. These are for example words that could either be a noun or a verb, depending on the context.

Syntactic performance is not only a desirable property if the tasks are syntactic. These properties also improve semantic prediction tasks. The role of a word has implications for the meaning of a word. It therefore is of academic and practical relevance to evaluate syntactic qualities of these representations, in an effort to improve the underlying models.

The experimental setup that has been developed can be used to evaluate syntactic properties of other models of obtaining sentence representations, such as the recently proposed CNN-LSTM model (Gan et al, 2017). Future work could focus on standardising an evaluation task, comparable to the tasks that are used in evaluation of semantic properties.

2 Related work

2.1 Syntax in natural language

One of the most prevailing definition of syntax in sentences is that it studies the principles and processes by which sentences are constructed (Chomsky, 1957). A related concept is that of grammaticality; the idea that a sentence can be grammatically correct, without containing any semantic information (Chomsky, 1957). This point is generally illustrated by making sentences with words that have no meaning. Despite the lack of meaning of the individual words, such sentences can still be considered grammatically correct by native speakers of the language.

Grammaticality does not apply in the other direction however. Grammatical properties of words are essential in determining the meaning of a sentence.

Information about syntax can also be enclosed at the word level. For many words, its basic grammatical function (e.g. noun or verb) can be deduced from the word alone. Some words are more ambiguous, and require syntactic context to be able to determine the grammatical function. Grammatical functions that depend more heavily on sentence structure (e.g. object or subject) can often not be deduced from the word alone. In the experiments, this difference has been observed between part-of-speech and dependency tags.

2.2 Sentence representations

Since some of the classic work showed that distributed representations of language offer significant benefits over localist approaches of encoding language (Elman, 1991), lots of progress has been

made. Several unsupervised approaches performed well (Le, Mikolov, 2014; Kiros et al, 2015; Gan et al, 2017).

Sentence representations with a distributed nature offer two major advantages. Firstly, relative temporal positions are encoded similarly, despite any offsets (Elman, 1990). Representations are therefore less sensitive to absolute position, which is important in encoding language. Secondly, distributed representations allow for fixed-length vectors (Elman, 1990).

Common approaches of representing sentences are based on the frequency of words (bag-of-words), sometimes adjusted for frequency in the entire dataset (tf-idf). The main disadvantage of this kind of approach is the lack of word order in the vector representing the sentence. A strength that should not be overlooked is its simplicity.

The strong suit of unsupervised setups, as opposed to supervised setups, is the generic nature of the resulting representations. Supervised setups tend to produce sentence representations that are more limited in applicability. Evaluation of supervised setups is more straightforward however. The same metric that has been used in training the model can be used to evaluate it.

Inspired by the skip-thoughts model, Gan et al (2017)have used a convolutional neural network as an encoder, instead of an RNN. Rather than predicting two surrounding sentences in training, Gan et al (2017) optimise on predicting multiple future sentences. Finally, in a composite model, the authors combine an autoencoder with a future sentence predictor.

Many of the unsupervised models are autoencoders. Besides autoencoders, more complex encoder-decoder setups have been used succesfully; examples are CNN-LSTM (Gan et al, 2017), RNN-RNN (Kiros et al, 2015) and even RNN-CNN (Tang et al, 2017) setups. Besides experimentation with network architecture, researchers search to adapt cost functions to improve hidden states in encoder-decoder models.

Recent work showed that supervised methods of learning sentence representations also have a place in the landscape of natural language processing, and tried to offer a solution for stagnating progress in unsupervised methods (Conneau et al, 2017; Socher et al, 2013). Conneau et al (2017) developed a supervised setup based on predicting Natural Language Inference tags.

$$\sum_t \log P(w_{i+1}^t | w_{i+1}^{<t}, h_i) + \sum_t \log P(w_{i-1}^t | w_{i-1}^{<t}, h_i)$$

Figure 1: The skip-thoughts loss function, adapted from Kiros et al (2015)

Distributed representations do have drawbacks however. It is for example not possible to directly link a logical grammatical function to a single scalar in the vector, although these effects can sometimes be observed (Kádár et al, 2016).

The skip-thoughts model, an RNN-RNN encoder-decoder model, optimises the hidden state h_i^t to predict the words in the sentences surrounding a sentence (see Figure 1). Therefore, to train the skip-thoughts model, a continuous body of text is required. Kiros et al (2015) used the BookCorpus (Zhu et al, 2015) for this.

The skip-thoughts model learns sentence representations from the sentences that form the context. This method has some similarity to the original method of learning word representations (Mikolov et al, 2013). The underpinning notion of learning from context has been transferred to the domain of sentence representations.

The skip-thoughts model uses Recurrent Neural Networks with Gated Recurrent Units for both the encoder and decoder part of the setup. Two variants of the skip-thoughts model exist: uni-skip and bi-skip. As the name suggest, uni-skip is an encoder that preserves sentence order (unidirectional). Bi-skip uses two encoders, one of which is given the sentence in reverse order. Since the encoders iterate over the words in the sentence, it is possible to use the intermediate hidden states (h_i^t) for the experiments in this study.

Both variants of the model have two RNN decoders: one for predicting the previous and one for predicting the next sentence (s_{i-1}, s_{i+1}). The combination of these two variants, used in this study, is called combine-skip. It simply is a concatenation of the vectors produced by uni-skip and bi-skip.

Finally, to make the model more generally usable,

Kiros et al (2015) train a linear mapping between the vast set of words that appear in word2vec and the more limited set of words used in training. Since the same BookCorpus is used in this study's experiments, this vocabulary expansion has not been applied.

2.3 Evaluating sentence representations

Evaluations of sentence representations are mostly semantic artificial prediction tasks. The skip-thoughts model has been evaluated on tasks like image retrieval, semantic similarity scoring and paraphrase detection (Kiros et al, 2015). Le, Mikolov (2014) evaluate their model using sentiment analysis on the Stanford Treebank and IMDB datasets.

Although, evaluation tasks of sentence representations are predominantly semantic, recent research shows some interesting new ideas for evaluating sentence representations. Li et al (2015) show that RNNs to generate sentence embeddings pay selective attention to words that are important for the prediction task. For this, the authors use saliency scores for words, as a way of measuring the influence of a single word towards the final classification. This is achieved by training a classifier that learns saliency scores for combinations of input and labels.

A similar approach to estimate salience of words involves omitting tokens from the sentence, and estimating the effect this incurs (Kádár et al, 2016). This has been done by measuring the distance between sentence representations with the word included and omitted. Among other observations, this experiment showed that the evaluated IMAGINET (Chrupała et al, 2015) model is sensitive to the grammatical role of words: "The image prediction pathway (...) learns to treat the same input token differently depending on its grammatical functions in the sentence (Kádár et al, 2016)".

The strategy of omission has also been applied succesfully by Li et al (2016). Besides estimating the importance of a certain word in the sentence, Li et al (2016) estimate the importance of a dimension in a vector. In short, Li et al (2016) calculate importance as the relative difference between the log-likelihoods of the correct class, with and without one vector dimension omitted. Omission is not used as an evaluation technique in the current study, since it would break the grammatical structure of the sentence, likely reducing the quality of all (intermediate) hidden states.

Several of these evaluations conclude that setups for obtaining sentence representations learn syntactic properties; even if the evaluation task is semantic (Kádár et al, 2016; Li et al, 2016; Li et al, 2015).

Strictly syntactic evaluation tasks are less commonplace. Classic work by Elman (1991) has a strong syntactic focus, as the simple recurrent network is trained to predict the next word in the sequence. This predictive ability of the model is evidence that abstract structure is encoded in internal representation (Elman, 1991). However, this approach is not transferable to the distributed sentence representations of today.

Adi et al (2017) recently proposed a methodology to evaluate syntactic properties of sentence representations. The main idea is to adopt artificial prediction tasks to evaluate sentence representations. Adi et al (2017) have opted to predict sentence length, word content and word order. A surprising result is that an encoder-decoder's ability to recreate sentences (the BLEU task) does not necessarily mean it will do well on other artificial prediction tasks. This finding underlines the need for more syntactic prediction tasks.

The aformentioned related work shows a need for syntactic evaluation metrics. Saliency or importance scores have not been used in the newly developed experimental setup, although there is a special focus on the word-level of sentence representations. A unique element of this study is the use of the intermediate hidden states of sentence representations at one word in the sentence, instead of using the final hidden state of an encoder-decoder model as a sentence representation. Finally, the idea of formulating artificial prediction tasks to compare models has been incorporated in this study's experimental setup.

3 Experimental setup

3.1 Experiments to measure syntactic properties

To estimate the syntactic properties, an artificial prediction task has been conceived, as suggested by Adi et al (2017). The artificial task that has been formulated is predicting grammatical categories of words. The grammatical categories that have been chosen

are part-of-speech (POS) tags and dependency tags (Nivre et al, 2017). To do so, a logistic regression classifier has been optimised. The accuracy that can be achieved gives an indication of the syntactic properties of the sentence representations. It could allow for future comparison of several models.

Other classifiers, such as a multilayer perceptron, would probably be able to achieve better results. It would however make it harder to separate properties of the vectors from abstraction abilities of the algorithm. This consideration has led to the choice of logistic regression.

3.2 Hyperparameter search

The hyperparameters have been tuned by searching a grid (see Table 1).

Hyperparameter	Set of values
Regularisation method	L1, L2
Inverse regularisation strength (C)	0.8, 1.0, 1.2, 2
Class weight	Balanced (adjusting weights inversely proportional to class frequencies), unbalanced

Table 1: Hyperparameters that have been optimised (cross-validated grid search)

Some of the best hyperparameters were found at the extreme ends of the search grid. This means better parameters are likely to be found.

3.3 Data

The training dataset consists of 10,000 sentences randomly sampled from the BookCorpus (2015). These sentences contain a total of 131,200 words, 119,282 of which are subsequent to the previous word. The test dataset consists of 2,500 sentences, 32,886 words, of which 29,881 are subsequent to the previous word.

This corpus already had tokenisation applied by NLTK (Bird et al, 2009). To prepare the corpus for the experiments, the hidden states for the words of the model have been generated with the skip-thoughts model, word embeddings have been looked up in the unidirectional skip-thoughts model and the target POS and dependency tags have been generated.

3.4 Feature vectors

In the prediction task, four types of feature vectors have been used. The first consisted of word embeddings from the skip-thoughts model for a single word (w_i^t). The second was similar, but with the word embedding of the previous word concatenated (w_i^t, w_i^{t-1}). Finally, the third extended this with the word embedding for the subsequent word ($w_i^t, w_i^{t-1}, w_i^{t+1}$). These types of concatenations require that there are previous and subsequent words within the same sentence. For first and last words of the sentence, the feature vectors have been zero-padded in the respective locations.

The final type of feature vectors is central to the evaluation of the skip-thoughts model. These are the hidden states of the skip-thoughts model, at the position of a word (h_i^t).

3.5 Target labels

The target labels for the multi-class classification task have been obtained by using SpaCy (2017a). The sentences of BookCorpus have been annotated by the models included in this software. For each word in the dataset, a part-of-speech tag and a dependency tag has been generated, with the `en_core_web_sm` model developed by SpaCy. These serve as a proxy for ground-truth data. Accuracy of these tags is reported at 97.04% for part-of-speech tags, and at 89.80% for dependency tags (SpaCy, 2017b).

3.6 Repeating experiments

All code to repeat the experiments in this experimental setup has been uploaded to an online repository[1]. This repository uses scikit-learn's (Pedregosa et al, 2011) implementation of logistic regression. Besides that, the Theano (2016) implementation of the skip-thoughts model was adapted to produce the intermediate hidden states. This adaptation has also been uploaded[2].

[1] https://github.com/bartbroere/syntactic-properties-of-skip-thought-vectors/

[2] https://github.com/bartbroere/skip-thoughts/

The same conventions for notation as Kiros et al (2015) use, are used throughout this thesis

h is the hidden state

w is the word

s is the range of words in the sentence.

Subscript i is the position of the sentence.

Superscript t is the position of the word.

4 Results

Isolating grammatical effects caused by syntax from those caused by syntactic properties at the word-level resulted in a few observations. First of all, predicting dependency tags is a harder task than predicting POS tags, as expected. The primary cause for this is the higher number of classes in this task. Grammatical function in dependency graphs is also more fine-grained than in POS tags; one word could potentially be placed in a wider range of classes.

Table 2 shows that the skip-thoughts hidden states offer an advantage over word embeddings, mostly in predicting dependency tags. The table also shows that performance with word embeddings overtakes the hidden states classifier, if previous and subsequent word embeddings are included in the feature vectors. For most real-life applications however, where sentences are instances, concatenating word embeddings would not be a viable option, since this results in variable-sized feature vectors.

Since the differences between the classifiers are larger in predicting dependency tags, the following section will focus on the reasons for this. Table 3 shows the accuracies in predicting dependency tags, achieved by the logistic regression optimised on several of the feature vectors. The columns with RER show the Relative Error Reduction that the hidden states classifier achieves over the three classifiers optimised on word embeddings.[3]

Table 3 shows that the hidden states classifier mostly does not outperform the classifier with three concatenated word embeddings. Two significant cases where the hidden states classifier did outperform the word embeddings however, are among others the classification of the root of the sentence (dependency tag `ROOT`), and the classification of objects of prepositions (dependency tag `pobj`). This could indicate improved syntactic sensitivity, since these are highly grammatically dependent tags.

[3]To save time in computation, the experiments with two and three word embeddings have been repeated with an optimal set of hyperparameters found in a previous, similar experiment. These experiments therefore have no standard deviation of cross validated accuracy in Table 2.

A selection of verbs (`VERB`) in their context, that the word embeddings classifier tagged incorrectly, but the hidden states classifier tagged correctly
we have developed this event to directly **address** the issues unique to earth .
i **answer** it to find the army nurse
i **glance** in my rearview mirror to see sawyer and ryan - our security for the day - climb into the audi suv .
more likely it will just **piss** him off .
youre beautiful , like he said , and surprisingly he **scratches** his head .

Figure 2: Differences in classification of verbs between the single word embedding and hidden states classifiers, in the part-of-speech task

Looking at the errors in prediction on a test set also revealed some patterns. The dependency tag classifier tends to overfit on frequent classes like the subject and root of a sentence. These are classes that are present in every correct sentence. The POS tag classifier displays the same behaviour, but with nouns and verbs.

Selecting the words that the hidden states classifier predicted correctly, but the single word embeddings model misclassified, confirms the hypothesis that syntactic context is present in skip-thought vectors. Figure 2 shows that this set of words contains mostly ambiguous words, where the sentence structure is essential in determining its grammatical function: *address, answer, glance, piss, scratches*.

Figure 3 shows that the same is the case for nouns. Instances where the hidden states classifier outperforms the word embeddings baseline are mostly ambiguous words, like: *fight, hope, look, study, will*.

4.1 Limitations

One major limitation of the experiments that have been conducted is the lack of ground-truth data, instead of the used close-to-ground-truth data. It is however expected that the results are similar if ground-truth data would have been used, since

Feature vectors	Target labels	Accuracy (standard deviation in cross-validation)	Accuracy (test set)
Word embeddings: w_i^t	POS tags	0.9323 (0.0006)	0.9340
Word embeddings: w_i^t w_i^{t-1}	POS tags	0.9595	0.9533
Word embeddings: w_i^t w_i^{t-1} w_i^{t+1}	POS tags	0.9794	0.9688
Hidden states: h_i^t	POS tags	0.9460 (0.0006)	0.9497
Word embeddings: w_i^t	Dependency tags	0.6979 (0.0011)	0.6990
Word embeddings: w_i^t w_i^{t-1}	Dependency tags	0.7886	0.7659
Word embeddings: w_i^t w_i^{t-1} w_i^{t+1}	Dependency tags	0.8490	0.8206
Hidden states: h_i^t	Dependency tags	0.8169 (0.0011)	0.8218

Table 2: Scores of the most performant logistic regression models

A selection of nouns (NOUN) in their context, that the word embeddings classifier tagged incorrectly, but the hidden states classifier tagged correctly
this is not your **fight** .
there was absolutely no **hope** .
he gave her an impressed **look** .
in a cluttered **study** carrel on the fifth level of the law library , between the racks of thick , seldom-used law books , darby shaw scanned a printout of the supreme court 's docket .
i sat on the chopping block , lacking the **will** to rise .

Figure 3: Differences in classification of nouns between the single word embedding and hidden states classifiers, in the part-of-speech task

SpaCy's tags are reported to quite accurate (SpaCy, 2017b).

Unfortunately, the BookCorpus has not been divided in a train, development and test set. All data has been used in training the skip-thoughts model. This means that the data used in this evaluation is technically not unseen data. The prediction task, however, is not the same, making it an allowable compromise.

Although the sequential nature is a strength of the skip-thoughts model in many respects, it poses a limitation to the isolation of the experiments in this thesis. The hidden states at the word level h_i^t do not isolate the latest processed word (t). (Kiros et al, 2015) state that h_i^t "can be interpreted as the representation of the sequence w_i^1 ... w_i^t". To predict syntactic categories, a classifier has to isolate the current word from this representation of a sequence.

Yet, despite this potential handicap of the data, the hidden state classifier outperforms the single word embeddings classifier. Looking at the instances where the hidden state classifier outperformed the word embedding classifier indicated that these are mostly ambiguous words. The context that is required for correct classification of these ambiguous words can be provided by either surrounding word embeddings, or by the hidden states of the skip-thoughts model.

The experiments with the task of predicting dependency tags also suffer from the drawback that the data limits the task to predicting the dependency tag, but not the dependency arc.

The limitations mentioned here open several avenues towards future work, which will be discussed in the next section.

4.2 Future work

There are two directions, although not mutually exclusive, future work could take to improve sentence representations. The first direction would obey the philosophy of doing one thing and doing it well. This direction would most likely result in ever more accurate POS-tagging models and better semantic sentence representations. This approach has the flaw that several applications could benefit from the combination of these models, and would have to incorporate several models.

The second direction is a more holistic approach, that continues to try to optimise sentence representations for better performance at both syntactic and

Dependency tag	Count	Word embeddings: w_i^t	Word embeddings: $w_i^t\ w_i^{t-1}$	Word embeddings: $w_i^t\ w_i^{t-1}\ w_i^{t+1}$	Hidden states: h_i^t	RER over w_i^t	RER over $w_i^t\ w_i^{t-1}$	RER over $w_i^t\ w_i^{t-1}\ w_i^{t+1}$
acl	171	0.05	0.30	0.28	0.20	15.43	-14.17	-11.38
acomp	347	0.49	0.80	0.80	0.78	57.30	-7.04	-11.76
advcl	716	0.16	0.20	0.22	0.59	51.25	49.04	47.48
advmod	1700	0.84	0.86	0.90	0.85	8.76	-8.23	-53.37
amod	919	0.75	0.82	0.88	0.81	25.43	-2.37	-61.68
appos	124	0.01	0.16	0.19	0.13	12.20	-3.85	-8.00
attr	253	0.02	0.22	0.25	0.55	54.44	42.64	40.21
aux	1514	0.88	0.86	0.95	0.84	-37.78	-16.98	-249.30
auxpass	150	0.04	0.11	0.81	0.17	13.89	6.77	-327.59
case	121	0.97	0.96	0.94	0.96	-25.00	0.00	28.57
cc	948	0.98	0.98	0.99	0.98	0.00	0.00	-7.14
ccomp	673	0.02	0.10	0.12	0.42	41.45	35.82	34.96
compound	411	0.15	0.15	0.72	0.09	-6.90	-6.90	-217.95
conj	961	0.02	0.43	0.44	0.66	65.60	40.84	39.85
det	2191	0.97	0.97	0.99	0.97	-1.56	9.72	-195.45
dobj	1897	0.36	0.55	0.63	0.73	57.86	40.68	26.87
intj	114	0.59	0.48	0.68	0.56	-6.38	15.25	-35.14
mark	499	0.82	0.78	0.84	0.78	-21.11	0.91	-34.57
neg	369	0.97	0.97	0.99	0.97	0.00	0.00	-450.00
npadvmod	139	0.27	0.44	0.51	0.41	18.81	-5.13	-20.59
nsubj	3472	0.77	0.81	0.90	0.88	48.64	35.86	-20.99
nsubjpass	106	0.00	0.00	0.03	0.00	0.00	0.00	-2.91
pcomp	118	0.14	0.62	0.59	0.38	28.43	-62.22	-52.08
pobj	2181	0.54	0.70	0.79	0.85	67.33	50.61	28.19
poss	1023	0.90	0.87	0.98	0.89	-11.54	12.78	-582.35
prep	2494	0.86	0.92	0.97	0.93	53.48	16.92	-101.20
prt	257	0.77	0.75	0.81	0.75	-6.78	1.56	-28.57
punct	5316	1.00	1.00	1.00	1.00	-50.00	-50.00	-50.00
relcl	325	0.00	0.19	0.20	0.41	40.92	27.27	26.15
ROOT	2454	0.70	0.73	0.76	0.81	36.19	28.64	20.00
xcomp	443	0.22	0.79	0.80	0.67	57.56	-53.68	-64.04

Table 3: Scores per class in predicting dependency tags, and relative error reduction over the hidden states classifier. Classes with less than 100 instances in the test set have been left out of this table.

semantic evaluation metrics. It is not unlikely that semantic performance will improve if syntactic relations are encoded better. The exploration of the field showed that successes have been booked with both supervised and unsupervised methods. Combining unsupervised and supervised elements in a single loss function, could result in even better sentence representations.

Future work in evaluating sentence representations that could elaborate on the findings in this thesis could try to predict complete dependency graphs from sentence representations, as an auxiliary prediction task. The practical implication of such experiments is that the experiments would also predict the dependency arcs, not only the tags. Devising an experimental setup for this is more challenging, but most likely would result in more detailed conclusions by better isolating syntax.

Simply training the classifiers with more instances is also likely to generate better results. This is possible: acquiring data is cheap because it can be generated with SpaCy. Adding more instances would probably eventually require an out-of-core experimental setup.

Repeating the experiments in this thesis with actual ground-truth data is also an interesting avenue of research. This would also bring a standardised syntactic evaluation task closer.

Similarly, applying this auxiliary prediction task to other methods of obtaining sentence representation could give more insight in what types of encoder-decoder setups perform well at encoding syntactic properties. It would also tell more about the feasibility of this evaluation setup.

5 Conclusion

On both tasks the hidden states classifier performed better than the single word embeddings classifier, showing that the syntax of the skip-thoughts hidden states offers an advantage over single word embeddings.

By using the artificial tasks of predicting POS and dependency tags, the presence of syntactic properties in skip-thought vectors has been explored. Logistic regression optimised on skip-thoughts hidden states is relatively successful in predicting grammatical functions of tokens. This is especially true if the grammatical function depends more heavily on syntax, as was the case in predicting some of the dependency tags.

Although similar performance could also be achieved by concatenating word embeddings, sentence representations offer the advantage of fixed-width vectors. In representing sentences, the length of concatenated word embeddings would depend on the length of the sentence, which is often not desirable.

Predicting grammatical functions of words from intermediate hidden states has proven to be a useful method of estimating syntax in sentence representations. This method has been applied to skip-thought vectors, showing that these vectors are relatively capable of predicting grammatical categories. To see if this method is suitable as an evaluation metric, it can be applied to other models in future work.

Acknowledgments

Firstly, I would like to thank Grzegorz Chrupała for his supervision and the courses he taught during the master. I am also grateful for the help of Lieke Gelderloos, who explained to me how I could extract intermediate hidden states using the reference implementation of the skip-thoughts model. Finally, I would especially like to thank my family for their support during my studies.

References

Adi, Y. & Kermany, E. & Belinkov, Y. & Lavi, O. & Goldberg, Y. (2017). *Fine-grained analysis of sentence embeddings using auxiliary prediction tasks.* In: ICLR 2017. Retrieved from: `https://arxiv.org/abs/1608.04207`

Bird, S. & Klein, E. & Loper, E. (2009). *Natural language processing with Python.* Retrieved from: `http://www.nltk.org/book/`

Chomsky, N. (1957). *Syntactic structure.* Berlin: Mouton de Gruyter

Chrupała, G. & Kádár, Á. & Alishahi, A. (2015). *Learning language through pictures.* In: ACL 2015. Retrieved from: `https://arxiv.org/pdf/1506.03694.pdf`

Conneau, A. & Kiela, D. & Schwenk, H. & Barrault, L. & Bordes, A. (2017). *Supervised learn-*

ing of universal sentence representations using natural language inference data. In: EMNLP 2017. Retrieved from: https://arxiv.org/pdf/1705.02364.pdf

De Marneffe, M.-C. & Manning, C.D. (2008). *Stanford typed dependencies manual.* Retrieved from: https://nlp.stanford.edu/software/dependencies_manual.pdf

Elman, J.L. (1990). *Finding structure in time.* In: Cognitive Science, issue 14, pp. 179-211.

Elman, J.L. (1991). *Distributed representations, simple recurrent networks, and grammatical structure.* In: Machine Learning, issue 7, pp. 195-225.

Gan, Z. & Pu, Y. & Henao, R. & Li, C. & He, X. & Carin, L. (2017). *Learning Generic Sentence Representations Using Convolutional Neural Networks.* In: EMNLP 2017, p. 2379-2389. Retrieved from: https://arxiv.org/abs/1611.07897

Goodfellow, I. & Bengio, Y. & Courville, A. (2016). *Deep learning* Cambridge: Massachusetts Institute of Technology.

Hill, F. & Cho, K. & Korhonen, A. (2016). *Learning distributed representations of sentences from unlabelled data.* In: Proceedings of NAACL-HLT 2016, pages 1367–1377. Retrieved from: https://arxiv.org/abs/1602.03483

Kádár, Á. & Chrupała, G. & Alishahi, A. (2016). *Representations of linguistic form and function in recurrent neural networks.* In: Computational Linguistics, 43(4):761-780. Retrieved from: https://arxiv.org/abs/1602.08952

Kiros, R. & Zhu, Y. & Salakhutdinov, R. & Zemel, R.S. & Torralba, A. & Urtasun, R & Fidler, S. (2015). *Skip-thought vectors.* In: NIPS, vol. 2, pp. 3294-3302. Retrieved from: https://arxiv.org/abs/1506.06726

Le, Q. & Mikolov, T. (2013) *Distributed representations of sentences and documents.* Retrieved from: https://arxiv.org/abs/1405.4053

Li, J. & Chen, X. & Hovy, E. & Jurafsky, D. (2015). *Visualizing and understanding neural models in NLP.* In: CoRR 2015. Retrieved from: https://arxiv.org/pdf/1506.01066.pdf

Li, J. & Monroe, W. & Jurafsky, D. (2016). *Understanding neural networks through representation erasure.* In: CoRR 2016. Retrieved from: http://arxiv.org/abs/1612.08220

Mikolov, T. & Sutskever, I. & Chen, K. & Corrado, G. & Dean, J. (2013). *Distributed representations of words and phrases and their compositionality.* Retrieved from: https://arxiv.org/abs/1310.4546

Nivre, J. et al (2017). *Universal dependencies (version 2.0).* Retrieved from: https://universaldependencies.org/

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

Socher, R. & Perelygin, A. & Wu, J.Y. & Chuang, J. & Manning, C.D. & Ng, A.Y. & Potts, C. (2013). *Recursive deep models for semantic compositionality over a sentiment treebank.* In: EMNLP 2013. Retrieved from: https://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf

SpaCy (2017a). *Industrial-Strength Natural Language Processing in Python.* Retrieved from: https://spacy.io/

SpaCy (2017b). *SpaCy: Available statistical models for English.* Retrieved from: https://spacy.io/models/en#en_core_web_sm

Tang, S. & Jin, H. & Fang, C. & Wang, Z. & De Sa, V.R. (2017). *Exploring asymmetric encoder-decoder structure for context-based sentence representation learning.* Retrieved from: https://arxiv.org/pdf/1710.10380.pdf

Theano Development Team (2016). *Theano: A Python framework for fast computation of mathematical expressions.*

Zhu, Y. & Kiros, R. & Zemel, R. & Salakhutdinov, R. & Urtasun, R. & Torralba, A. & Fidler, S. (2015). *Aligning books and movies: Towards story-like visual explanations by watching movies and reading books.* Retrieved from: `https://arxiv.org/abs/1506.06724`