# Let's do it "again": A First Computational Approach to Detecting Adverbial Presupposition Triggers

**Andre Cianflone** [*] **Yulan Feng** [*] **Jad Kabbara** [*] **Jackie Chi Kit Cheung**

School of Computer Science
McGill University
Montreal, QC, Canada

MILA
Montreal, QC, Canada

{andre.cianflone@mail, yulan.feng@mail, jad@cs, jcheung@cs}.mcgill.ca

## Abstract

We introduce the task of predicting adverbial presupposition triggers such as *also* and *again*. Solving such a task requires detecting recurring or similar events in the discourse context, and has applications in natural language generation tasks such as summarization and dialogue systems. We create two new datasets for the task, derived from the Penn Treebank and the Annotated English Gigaword corpora, as well as a novel attention mechanism tailored to this task. Our attention mechanism augments a baseline recurrent neural network without the need for additional trainable parameters, minimizing the added computational cost of our mechanism. We demonstrate that our model statistically outperforms a number of baselines, including an LSTM-based language model.

## 1 Introduction

In pragmatics, presuppositions are assumptions or beliefs in the common ground between discourse participants when an utterance is made (Frege, 1892; Strawson, 1950; Stalnaker, 1973, 1998), and are ubiquitous in naturally occurring discourses (Beaver and Geurts, 2014). Presuppositions underly spoken statements and written sentences and understanding them facilitates smooth communication. We refer to expressions that indicate the presence of presuppositions as *presupposition triggers*. These include definite descriptions, factive verbs and certain adverbs, among others. For example, consider the following statements:

(1) John is going to the restaurant *again*.

(2) John has been to the restaurant.

(1) is only appropriate in the context where (2) is held to be true because of the presence of the presupposition trigger *again*. One distinguishing characteristic of presupposition is that it is unaffected by negation of the presupposing context, unlike other semantic phenomena such as entailment and implicature. The negation of (1), *John is not going to the restaurant again.*, also presupposes (2).

Our focus in this paper is on adverbial presupposition triggers such as *again, also* and *still*. Adverbial presupposition triggers indicate the recurrence, continuation, or termination of an event in the discourse context, or the presence of a similar event. In one study of presuppositional triggers in English journalistic texts (Khaleel, 2010), adverbial triggers were found to be the most commonly occurring presupposition triggers after existential triggers.[1] Despite their frequency, there has been little work on these triggers in the computational literature from a statistical, corpus-driven perspective.

As a first step towards language technology systems capable of understanding and using presuppositions, we propose to investigate the detection of contexts in which these triggers can be used. This task constitutes an interesting testing ground for pragmatic reasoning, because the cues that are indicative of contexts containing recurring or similar events are complex and often span more than one sentence, as illustrated in Sentences (1) and (2). Moreover, such a task has immediate practical consequences. For example, in language generation applications such as summarization and dialogue systems, adding presuppositional triggers in contextually appropriate loca-

[*] Authors (listed in alphabetical order) contributed equally.

[1] Presupposition of existence are triggered by possessive constructions, names or definite noun phrases.

tions can improve the readability and coherence of the generated output.

We create two datasets based on the Penn Treebank corpus (Marcus et al., 1993) and the English Gigaword corpus (Graff et al., 2007), extracting contexts that include presupposition triggers as well as other similar contexts that do not, in order to form a binary classification task. In creating our datasets, we consider a set of five target adverbs: *too, again, also, still*, and *yet*. We focus on these adverbs in our investigation because these triggers are well known in the existing linguistic literature and commonly triggering presuppositions. We control for a number of potential confounding factors, such as class balance, and the syntactic governor of the triggering adverb, so that models cannot exploit these correlating factors without any actual understanding of the presuppositional properties of the context.

We test a number of standard baseline classifiers on these datasets, including a logistic regression model and deep learning methods based on recurrent neural networks (RNN) and convolutional neural networks (CNN).

In addition, we investigate the potential of attention-based deep learning models for detecting adverbial triggers. Attention is a promising approach to this task because it allows a model to weigh information from multiple points in the previous context and infer long-range dependencies in the data (Bahdanau et al., 2015). For example, the model could learn to detect multiple instances involving *John* and *restaurants*, which would be a good indication that *again* is appropriate in that context. Also, an attention-based RNN has achieved success in predicting article definiteness, which involves another class of presupposition triggers (Kabbara et al., 2016).

As another contribution, we introduce a new weighted pooling attention mechanism designed for predicting adverbial presupposition triggers. Our attention mechanism allows for a weighted averaging of our RNN hidden states where the weights are informed by the inputs, as opposed to a simple unweighted averaging. Our model uses a form of self-attention (Paulus et al., 2018; Vaswani et al., 2017), where the input sequence acts as both the attention mechanism's query and key/value. Unlike other attention models, instead of simply averaging the scores to be weighted, our approach aggregates (learned) attention scores by learning a reweighting scheme of those scores through another level (dimension) of attention. Additionally, our mechanism does not introduce any new parameters when compared to our LSTM baseline, reducing its computational impact.

We compare our model using the novel attention mechanism against the baseline classifiers in terms of prediction accuracy. Our model outperforms these baselines for most of the triggers on the two datasets, achieving 82.42% accuracy on predicting the adverb "also" on the Gigaword dataset.

The contributions of this work are as follows:

1. We introduce the task of predicting adverbial presupposition triggers.

2. We present new datasets for the task of detecting adverbial presupposition triggers, with a data extraction method that can be applied to other similar pre-processing tasks.

3. We develop a new attention mechanism in an RNN architecture that is appropriate for the prediction of adverbial presupposition triggers, and show that its use results in better prediction performance over a number of baselines without introducing additional parameters.

## 2 Related Work

### 2.1 Presupposition and pragmatic reasoning

The discussion of presupposition can be traced back to Frege's work on the philosophy of language (Frege, 1892), which later leads to the most commonly accepted view of presupposition called the Frege-Strawson theory (Kaplan, 1970; Strawson, 1950). In this view, presuppositions are preconditions for sentences/statements to be true or false. To the best of our knowledge, there is no previous computational work that directly investigates adverbial presupposition. However in the fields of semantics and pragmatics, there exist linguistic studies on presupposition that involve adverbs such as "too" and "again" (e.g., (Blutner et al., 2003), (Kang, 2012)) as a pragmatic presupposition trigger. Also relevant to our work is (Kabbara et al., 2016), which proposes using an attention-based LSTM network to predict noun phrase definiteness in English. Their work demonstrates the ability of these attention-based models to pick up on contextual cues for pragmatic reasoning.

Many different classes of construction can trigger presupposition in an utterance, this includes but is not limited to stressed constituents, factive verbs, and implicative verbs (Zare et al., 2012). In this work, we focus on the class of adverbial presupposition triggers.

Our task setup resembles the Cloze test used in psychology (Taylor, 1953; E. B. Coleman, 1968; Earl F. Rankin, 1969) and machine comprehension (Riloff and Thelen, 2000), which tests text comprehension via a fill-in-the-blanks task. We similarly pre-process our samples such that they are roughly the same length, and have equal numbers of negative samples as positive ones. However, we avoid replacing the deleted words with a blank, so that our model has no clue regarding the exact position of the possibly missing trigger. Another related work on the Children's Book Test (Hill et al., 2015) notes that memories that encode sub-sentential chunks (windows) of informative text seem to be most useful to neural networks when interpreting and modelling language. Their finding inspires us to run initial experiments with different context windows and tune the size of chunks according to the Logistic Regression results on the development set.

### 2.2 Attention

In the context of encoder-decoder models, attention weights are usually based on an energy measure of the previous decoder hidden state and encoder hidden states. Many variations on attention computation exist. Sukhbaatar et al. (2015) propose an attention mechanism conditioned on a query and applied to a document. To generate summaries, Paulus et al. (2018) add an attention mechanism in the prediction layer, as opposed to the hidden states. Vaswani et al. (2017) suggest a model which learns an input representation by self-attending over inputs. While these methods are all tailored to their specific tasks, they all inspire our choice of a self-attending mechanism.

## 3 Datasets

### 3.1 Corpora

We extract datasets from two corpora, namely the Penn Treebank (PTB) corpus (Marcus et al., 1993) and a subset (sections 000-760) of the third edition of the English Gigaword corpus (Graff et al., 2007). For the PTB dataset, we use sections 22 and 23 for testing. For the Gigaword corpus, we use sections 700-760 for testing. For the remaining data, we randomly chose 10% of them for development, and the other 90% for training.

For each dataset, we consider a set of five target adverbs: *too, again, also, still*, and *yet*. We choose these five because they are commonly used adverbs that trigger presupposition. Since we are concerned with investigating the capacity of attentional deep neural networks in predicting the presuppositional effects in general, we frame the learning problem as a binary classification for predicting the presence of an adverbial presupposition (as opposed to the identity of the adverb).

On the Gigaword corpus, we consider each adverb separately, resulting in five binary classification tasks. This was not feasible for PTB because of its small size.

Finally, because of the commonalities between the adverbs in presupposing similar events, we create a dataset that unifies all instances of the five adverbs found in the Gigaword corpus, with a label "1" indicating the presence of any of these adverbs.

```
('still',
['The', 'Old', 'Granary', .../* 46
    tokens omitted */...,'has', '@@@@',
    'included', 'Bertrand', 'Russell',
    .../* 6 tokens omitted */... 'Morris
    '],
['DT', 'NNP', 'NNP', .../* 46 tokens
    omitted */..., 'VBZ', '@@@@', 'VBN',
     'NNP', 'NNP', .../* 6 tokens
    omitted */... 'NNP'])
```

Figure 1: An example of an instance containing a presuppositional trigger from our dataset.

### 3.2 Data extraction process

We define a sample in our dataset as a 3-tuple, consisting of a label (representing the target adverb, or 'none' for a negative sample), a list of tokens we extract (before/after the adverb), and a list of corresponding POS tags (Klein and Manning, 2002). In each sample, we also add a special token "@@@@" right before the head word and the corresponding POS tag of the head word, both in positive and negative cases. We add such special tokens to identify the candidate context in the passage to the model. Figure 1 shows a single positive sample in our dataset.

We first extract positive contexts that contain a triggering adverb, then extract negative contexts that do not, controlling for a number of potential confounds. Our positive data consist of cases where the target adverb triggers presupposition by modifying a certain head word which, in most cases, is a verb. We define such head word as a *governor* of the target adverb.

When extracting positive data, we scan through all the documents, searching for target adverbs. For each occurrence of a target adverb, we store the location and the governor of the adverb. Taking each occurrence of a governor as a pivot, we extract the 50 unlemmatized tokens preceding it, together with the tokens right after it up to the end of the sentence (where the adverb is)–with the adverb itself being removed. If there are less than 50 tokens before the adverb, we simply extract all of these tokens. In preliminary testing using a logistic regression classifier, we found that limiting the size to 50 tokens had higher accuracy than 25 or 100 tokens. As some head words themselves are stopwords, in the list of tokens, we do not remove any stopwords from the sample; otherwise, we would lose many important samples.

We filter out the governors of “too" that have POS tags “JJ” and “RB” (adjectives and adverbs), because such cases corresponds to a different sense of “too” which indicates excess quantity and does not trigger presupposition (e.g., “rely too heavily on”, “it's too far from”).

After extracting the positive cases, we then use the governor information of positive cases to extract negative data. In particular, we extract sentences containing the same governors but not any of the target adverbs as negatives. In this way, models cannot rely on the identity of the governor alone to predict the class. This procedure also roughly balances the number of samples in the positive and negative classes.

For each governor in a positive sample, we locate a corresponding context in the corpus where the governor occurs without being modified by any of the target adverbs. We then extract the surrounding tokens in the same fashion as above. Moreover, we try to control position-related confounding factors by two randomization approaches: 1) randomize the order of documents to be scanned, and 2) within each document, start scanning from a random location in the document. Note that the number of negative cases might not be exactly equal to the number of negative cases in all datasets because some governors appearing in positive cases are rare words, and we're unable to find any (or only few) occurrences that match them for the negative cases.

## 4 Learning Model

In this section, we introduce our attention-based model. At a high level, our model extends a bidirectional LSTM model by computing correlations between the hidden states at each timestep, then applying an attention mechanism over these correlations. Our proposed weighted-pooling (WP) neural network architecture is shown in Figure 2.

The input sequence $u = \{u_1, u_2, \dots, u_T\}$ consists of a sequence, of time length $T$, of one-hot encoded word tokens, where the original tokens are those such as in Listing 1. Each token $u_t$ is embedded with pretrained embedding matrix $W_e \in \mathbb{R}^{|V|\times d}$, where $|V|$ corresponds to the number of tokens in vocabulary $V$, and $d$ defines the size of the word embeddings. The embedded token vector $x_t \in \mathbb{R}^d$ is retrieved simply with $x_t = u_t W_e$. Optionally, $x_t$ may also include the token's POS tag. In such instances, the embedded token at time step $t$ is concatenated with the POS tag's one-hot encoding $p_t$: $x_t = u_t W_e || p_t$, where $||$ denotes the vector concatenation operator.

At each input time step $t$, a bi-directional LSTM (Hochreiter and Schmidhuber, 1997) encodes $x_t$ into hidden state $h_t \in \mathbb{R}^s$:

$$h_t = \left[\overrightarrow{h_t} || \overleftarrow{h_t}\right] \tag{1}$$

where $\overrightarrow{h_t} = f(x_t, h_{t-1})$ is computed by the forward LSTM, and $\overleftarrow{h_t} = f(x_t, h_{t+1})$ is computed by the backward LSTM. Concatenated vector $h_t$ is of size $2s$, where $s$ is a hyperparameter determining the size of the LSTM hidden states. Let matrix $H \in \mathbb{R}^{2s\times T}$ correspond to the concatenation of all hidden state vectors:

$$H = [h_1 || h_2 || \dots || h_T]. \tag{2}$$

Our model uses a form of self-attention (Paulus et al., 2018; Vaswani et al., 2017), where the input sequence acts as both the attention mechanism's query and key/value. Since the location of a presupposition trigger can greatly vary from one sample to another, and because dependencies can be long range or short range, we model all possible word-pair interactions within a sequence. We calculate the energy between all input tokens with a

| | Training set | | | Test set | | |
|---|---|---|---|---|---|---|
| Corpus | Positive | Negative | Total | Positive | Negative | Total |
| PTB | 2,596 | 2,579 | 5,175 | 249 | 233 | 482 |
| Gigaword yet | 32,024 | 31,819 | 63,843 | 7950 | 7890 | 15840 |
| Gigaword too | 55,827 | 29,918 | 85,745 | 13987 | 7514 | 21501 |
| Gigaword again | 43,120 | 42,824 | 85,944 | 10935 | 10827 | 21762 |
| Gigaword still | 97,670 | 96,991 | 194,661 | 24509 | 24232 | 48741 |
| Gigaword also | 269,778 | 267,851 | 537,626 | 66878 | 66050 | 132928 |
| Gigaword all | 498,415 | 491,173 | 989,588 | 124255 | 123078 | 247333 |

Table 1: Number of training samples in each dataset.

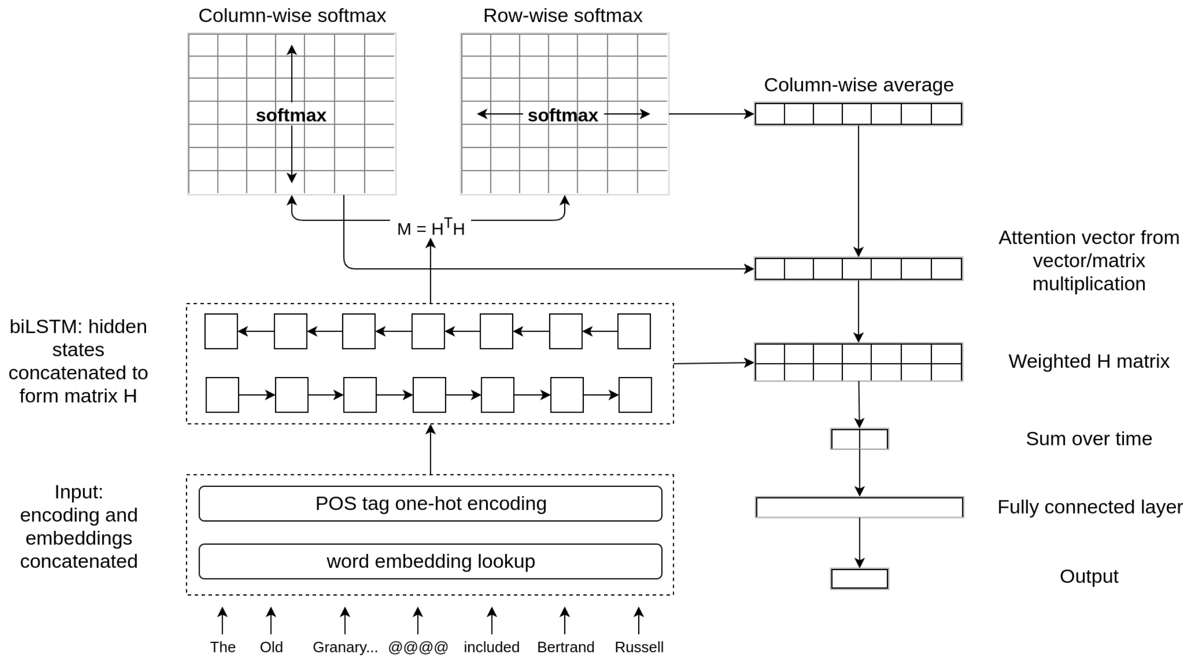

Figure 2: Our weighted-pooling neural network architecture (WP). The tokenized input is embedded with pretrained word embeddings and possibly concatenated with one-hot encoded POS tags. The input is then encoded with a bi-directional LSTM, followed by our attention mechanism. The computed attention scores are then used as weights to average the encoded states, in turn connected to a fully connected layer to predict presupposition triggering.

pair-wise matching matrix:

$$M = H^\top H \tag{3}$$

where $M$ is a square matrix $\in \mathbb{R}^{T\times T}$. To get a single attention weight per time step, we adopt the attention-over-attention method (Cui et al., 2017). With matrix $M$, we first compute row-wise attention score $M^r_{ij}$ over $M$:

$$M^r_{ij} = \frac{exp(e_{ij})}{\sum_{t=1}^{T} exp(e_{it})} \tag{4}$$

where $e_{ij} = M_{ij}$. $M^r$ can be interpreted as a word-level attention distribution over all other words. Since we would like a single weight per word, we need an additional step to aggregate these attention scores. Instead of simply averaging the scores, we follow (Cui et al., 2017)'s approach which learns the aggregation by an additional attention mechanism. We compute column-wise softmax $M^c_{ij}$ over $M$:

$$M^c_{ij} = \frac{exp(e_{ij})}{\sum_{t=1}^{T} exp(e_{tj})} \tag{5}$$

The columns of $M^r$ are then averaged, forming vector $\beta \in \mathbb{R}^T$. Finally, $\beta$ is multiplied with the column-wise softmax matrix $M^c$ to get attention vector $\alpha$:

$$\alpha = {M^r}^\top \beta. \tag{6}$$

Note Equations (2) to (6) have described how we derived an attention score over our input without the introduction of any new parameters, potentially minimizing the computational effect of our attention mechanism.

As a last layer to their neural network, Cui et al. (2017) sum over $\alpha$ to extract the most relevant input. However, we use $\alpha$ as weights to combine all of our hidden states $h_t$:

$$c = \sum_{t=1}^{T} \alpha_t h_t \tag{7}$$

where $c \in \mathbb{R}^s$. We follow the pooling with a dense layer $z = \sigma(W_z c + b_z)$, where $\sigma$ is a non-linear function, matrix $W_z \in \mathbb{R}^{64 \times s}$ and vector $b_z \in \mathbb{R}^{64}$ are learned parameters. The presupposition trigger probability is computed with an affine transform followed by a softmax:

$$\hat{y} = \text{softmax}(W_o z + b_o) \tag{8}$$

where matrix $W_o \in \mathbb{R}^{2 \times 64}$ and vector $b_o \in \mathbb{R}^2$ are learned parameters. The training objective minimizes:

$$J(\theta) = \frac{1}{m} \sum_{t=1}^{m} E(\hat{y}, y) \tag{9}$$

where $E(\cdot\, , \cdot)$ is the standard cross-entropy.

## 5 Experiments

We compare the performance of our WP model against several models which we describe in this section. We carry out the experiments on both datasets described in Section 3. We also investigate the impact of POS tags and attention mechanism on the models' prediction accuracy.

### 5.1 Baselines

We compare our learning model against the following systems. The first is the most-frequent-class baseline **(MFC)** which simply labels all samples with the most frequent class of 1. The second is a logistic regression classifier **(LogReg)**, in which the probabilities describing the possible outcomes of a single input $x$ is modeled using a logistic function. We implement this baseline classifier with the scikit-learn package (Pedregosa et al., 2011), with a CountVectorizer including bi-gram features. All of the other hyperparameters are set to default weights.

The third is a variant LSTM recurrent neural network as introduced in (Graves, 2013). The input is encoded by a bidirectional LSTM like the WP model detailed in Section 4. Instead of a self-attention mechanism, we simply mean-pool matrix $H$, the concatenation of all LSTM hidden states, across all time steps. This is followed by a fully connected layer and softmax function for the binary classification. Our WP model uses the same bidirectional LSTM as this baseline LSTM, and has the same number of parameters, allowing for a fair comparison of the two models. Such a standard LSTM model represents a state-of-the-art language model, as it outperforms more recent models on language modeling tasks when the number of model parameters is controlled for (Melis et al., 2017).

For the last model, we use a slight variant of the CNN sentence classification model of (Kim, 2014) based on the Britz tensorflow implementation[2].

### 5.2 Hyperparameters & Additional Features

After tuning, we found the following hyperparameters to work best: 64 units in fully connected layers and 40 units for POS embeddings. We used dropout with probability 0.5 and mini-batch size of 64.

For all models, we initialize word embeddings with word2vec (Mikolov et al., 2013) pretrained embeddings of size 300. Unknown words are randomly initialized to the same size as the word2vec embeddings. In early tests on the development datasets, we found that our neural networks would consistently perform better when fixing the word embeddings. All neural network performance reported in this paper use fixed embeddings.

Fully connected layers in the LSTM, CNN and WP model are regularized with dropout (Srivastava et al., 2014). The model parameters for these neural networks are fine-tuned with the Adam algorithm (Kingma and Ba, 2015). To stabilize the RNN training gradients (Pascanu et al., 2013), we perform gradient clipping for gradients below threshold value -1, or above 1. To reduce overfitting, we stop training if the development set does not improve in accuracy for 10 epochs. All performance on the test set is reported using the best trained model as measured on the development set.

In addition, we use the CoreNLP Part-of-

[2]http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/

| | | Accuracy | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | WSJ | Gigaword | | | | | |
| Models | Variants | All adverbs | All adverbs | Also | Still | Again | Too | Yet |
| MFC | - | 51.66 | 50.24 | 50.32 | 50.29 | 50.25 | 65.06 | 50.19 |
| LogReg | + POS | 52.81 | 53.65 | 52.00 | 56.36 | 59.49 | 69.77 | **61.05** |
| | - POS | 54.47 | 52.86 | 56.07 | 55.29 | 58.60 | 67.60 | 58.60 |
| CNN | + POS | 58.84 | 59.12 | 61.53 | 59.54 | 60.26 | 67.53 | 59.69 |
| | - POS | 62.16 | 57.21 | 59.76 | 56.95 | 57.28 | 67.84 | 56.53 |
| LSTM | + POS | 74.23 | 60.58 | 81.48 | 60.72 | **61.81** | **69.70** | 59.13 |
| | - POS | 73.18 | **58.86** | 81.16 | **58.97** | **59.93** | **68.32** | 55.71 |
| WP | + POS | **76.09** | **60.62** | **82.42** | **61.00** | 61.59 | 69.38 | 57.68 |
| | - POS | **74.84** | **58.87** | **81.64** | **59.03** | 58.49 | **68.37** | **56.68** |

Table 2: Performance of various models, including our weighted-pooled LSTM (WP). MFC refers to the most-frequent-class baseline, LogReg is the logistic regression baseline. LSTM and CNN correspond to strong neural network baselines. Note that we bold the performance numbers for the best performing model for each of the "+ POS" case and the "- POS" case.

Speech (POS) tagger (Manning et al., 2014) to get corresponding POS features for extracted tokens. In all of our models, we limit the maximum length of samples and POS tags to 60 tokens. For the CNN, sequences shorter than 60 tokens are zero-padded.

## 6 Results

Table 2 shows the performance obtained by the different models with and without POS tags. Overall, our attention model WP outperforms all other models in 10 out of 14 scenarios (combinations of datasets and whether or not POS tags are used). Importantly, our model outperforms the regular LSTM model without introducing additional parameters to the model, which highlights the advantage of WP's attention-based pooling method. For all models listed in Table 2, we find that including POS tags benefits the detection of adverbial presupposition triggers in Gigaword and PTB datasets. Note that, in Table 2, we bolded accuracy figures that were within 0.1% of the best performing WP model as McNemar's test did not show that WP significantly outperformed the other model in these cases ($p$ value > 0.05).

Table 3 shows the confusion matrix for the best performing model (WP,+POS). The small differences in the off-diagonal entries inform us that the model misclassifications are not particularly skewed towards the presence or absence of presupposition triggers.

| | | **Predicted** | |
|---|---|---|---|
| | | Absence | Presence |
| **Actual** | Absence | 54,658 | 11,961 |
| | Presence | 11,776 | 55,006 |

Table 3: Confusion matrix for the best performing model, predicting the presence of a presupposition trigger or the absence of such as trigger.

| | **WP Cor.** | **WP Inc.** |
|---|---|---|
| **LSTM Cor.** | 101,443 | 6,819 |
| **LSTM Inc.** | 8,016 | 17,123 |

Table 4: Contingency table for correct (cor.) and incorrect (inc.) predictions between the LSTM baseline and the attention model (WP) on the Giga_also dataset.

The contingency table, shown in Table 4, shows the distribution of agreed and disagreed classification.

## 7 Analysis

Consider the following pair of samples that we randomly choose from the PTB dataset (shortened for readability):

1. ...Taped just as the market closed yesterday , it offers Ms. Farrell advising , " We view

the market here as going through a relatively normal cycle ... . We **continue** to feel that the stock market is the @@@@ place to be for long-term appreciation

2. ...More people are remaining independent longer presumably because they are better off physically and financially . Careers count most for the well-to-do many affluent people @@@@ place personal success and money above family

In both cases, the head word is *place*. In Example 1, the word *continue* (emphasized in the above text) suggests that adverb *still* could be used to modify head word *place* (i.e., ... *the stock market is still the place* ...). Further, it is also easy to see that *place* refers to *stock market*, which has occurred in the previous context. Our model correctly predicts this sample as containing a presupposition, this despite the complexity of the coreference across the text.

In the second case of the usage of the same main head word *place* in Example 2, our model falsely predicts the presence of a presupposition. However, even a human could read the sentence as "many people still place personal success and money above family". This underlies the subtlety and difficulty of the task at hand. The long-range dependencies and interactions within sentences seen in these examples are what motivate the use of the various deep non-linear models presented in this work, which are useful in detecting these coreferences, particularly in the case of attention mechanisms.

## 8 Conclusion

In this work, we have investigated the task of predicting adverbial presupposition triggers and introduced several datasets for the task. Additionally, we have presented a novel weighted-pooling attention mechanism which is incorporated into a recurrent neural network model for predicting the presence of an adverbial presuppositional trigger. Our results show that the model outperforms the CNN and LSTM, and does not add any additional parameters over the standard LSTM model. This shows its promise in classification tasks involving capturing and combining relevant information from multiple points in the previous context.

In future work, we would like to focus more on designing models that can deal with and be optimized for scenarios with severe data imbalance. We would like to also explore various applications of presupposition trigger prediction in language generation applications, as well as additional attention-based neural network architectures.

## Acknowledgements

The authors would like to thank the reviewers for their valuable comments. This work was supported by the Centre de Recherche d'Informatique de Montréal (CRIM), the Fonds de Recherche du Québec – Nature et Technologies (FRQNT) and the Natural Sciences and Engineering Research Council of Canada (NSERC).

## References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Advances in Neural Information Processing Systems (NIPS 2015)*, pages 649–657, Montreal, Canada.

David I. Beaver and Bart Geurts. 2014. Presupposition. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, winter 2014 edition. Metaphysics Research Lab, Stanford University.

R. Blutner, H. Zeevat, K. Bach, A. Bezuidenhout, R. Breheny, S. Glucksberg, F. Happé, F. Recanati, and D. Wilson. 2003. *Optimality Theory and Pragmatics*. Palgrave Studies in Pragmatics, Language and Cognition. Palgrave Macmillan UK.

Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. 2017. Attention-over-attention neural networks for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 593–602.

G. R. Miller E. B. Coleman. 1968. A measure of information gained during prose learning. *Reading Research Quarterly*, 3(3):369–386.

Joseph W. Culhane Earl F. Rankin. 1969. Comparable cloze and multiple-choice comprehension test scores. *Journal of Reading*, 13(3):193–198.

Gottlob Frege. 1892. Über sinn und bedeutung. *Zeitschrift für Philosophie und philosophische Kritik*, 100:25–50.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2007. English gigaword third edition. Technical report, Linguistic Data Consortium.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2015. The goldilocks principle: Reading children's books with explicit memory representations. *CoRR*, abs/1511.02301.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Jad Kabbara, Yulan Feng, and Jackie Chi Kit Cheung. 2016. Capturing pragmatic knowledge in article usage prediction using lstms. In *COLING*, pages 2625–2634.

Qiang Kang. 2012. The use of too as a pragmatic presupposition trigger. *Canadian Social Science*, 8(6):165–169.

David Kaplan. 1970. What is Russell's theory of descriptions? In Wolfgang Yourgrau and Allen D. Breck, editors, *Physics, Logic, and History*, pages 277–295. Plenum Press.

Layth Muthana Khaleel. 2010. An analysis of presupposition triggers in english journalistic texts. *Of College Of Education For Women*, 21(2):523–551.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceeding of the 2015 International Conference on Learning Representation (ICLR 2015)*, San Diego, California.

Dan Klein and Christopher D. Manning. 2002. A generative constituent-context model for improved grammar induction. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 128–135, Stroudsburg, PA, USA. Association for Computational Linguistics.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Mitchell P. Marcus, Mary A. Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Gábor Melis, Chris Dyer, and Phil Blunsom. 2017. On the state of the art of evaluation in neural language models. *CoRR*, abs/1707.05589.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages 1310–1318.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*, Vancouver, Canada.

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

Ellen Riloff and Michael Thelen. 2000. A rule-based question answering system for reading comprehension tests. In *Proceedings of the 2000 ANLP/NAACL Workshop on Reading Comprehension Tests As Evaluation for Computer-based Language Understanding Sytems - Volume 6*, ANLP/NAACL-ReadingComp '00, pages 13–19, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958.

Robert Stalnaker. 1973. Presuppositions. *Journal of philosophical logic*, 2(4):447–457.

Robert Stalnaker. 1998. On the representation of context. *Journal of Logic, Language and Information*, 7(1):3–19.

Peter F. Strawson. 1950. On referring. *Mind*, 59(235):320–344.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems 28*, pages 2440–2448.

Wilson L. Taylor. 1953. Cloze procedure: A new tool for measuring readability. *Journalism Quarterly*, 30(4):415.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Llion Jones, Jakob Uszkoreit, Aidan N Gomez, and Ł ukasz Kaiser. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5994–6004.

Javad Zare, Ehsan Abbaspour, and Mahdi Rajaee Nia. 2012. Presupposition trigger—a comparative analysis of broadcast news discourse. *International Journal of Linguistics*, 4(3):734–743.