Graph-based Dependency Parsing to Improve Co-reference Resolution on the WinoGrande Challenge

Jack Zhang*
University of California, Los Angeles
dvorjackz@ucla.edu

Edward Zhang*
University of California, Los Angeles
edzhang@ucla.edu

Hirday Gupta*
University of California, Los Angeles
hirday@ucla.edu

Abstract

In our attempt to tackle the WinoGrande challenge, we categorically analyze the various approaches by past leaders, and propose a novel approach which borrows from other domains. We observe that high performing WinoGrande often use a mix of classification and language modelling techniques. We also notice that highly achieving BERT-based models in other domains have used the CLS token to make sentence-level classifications. This token captures sentence-level features, and we propose to enrich its sentence-representational power by adding dependency information.

Our upgraded CLS layer is able to achieve (54.4%) AUC on the WinoGrande challenge, placing us within the top 20 submissions and beating most baseline models. Our results suggest that there are features that may be hard for the CLS token represent. We also find that adding sentence-level features can significantly improve classification performance, in some cases that improvement does not match it up with the performance of language model approaches.

1 Introduction

The WinoGrande challenge is both a common sense reasoning task and a natural language understanding task. Each of the 44000 problems in the WinoGrande dataset is formulated as a sentence with an ambiguous pronoun and two choices (10). The task for the model is to choose the correct choice that the pronoun refers to.

> Sentence: "**Adam** was cold and **Michael** was hot so he removed a heavy jacket to withstand the freezing weather."
> Option 1: "Adam"
> Option 2: "Michael"

BERT and its variations have been a staple in the WinoGrande landscape. They output feature-rich encodings of each word in the input, as well as for a CLS token that is artificially inserted at the start of each sentence (10). The CLS token is able to capture information about the entire sentence and thus provide contextual insight (1). This idea has been used to great success in tasks that require tacit knowledge such as WinoGrande.

2 Related Work

2.1 Masked Language Modeling

Masked language modeling is a task which asks models to predict a masked word, given the masked word's context in a sentence. This was one of the primary tasks that was used to pre-train the BERT model (1), and was the primary task used to pre-train the RoBERTa model (6).

Thus, WinoGrande problems can be formulated as masked language modeling problems by simply replacing the ambiguous pronoun with a mask token and comparing the probability of each of the candidate nouns replacing the mask. The BERT and RoBERTa models can be minimally modified to solve these problems.

In particular, RoBERTa was able to achieve an accuracy of 78.1% on the WinoGrande dataset (6), which we were able to replicate using their codebase (8).

2.2 Text-to-Text Transformer and QA

There have been multiple examples of text-to-text transformer models being applied to the WinoGrande dataset. Lin et al use a formulation of the problem that is similar to a natural language inference (NLI) problem. (5) In this solution, the sentence of interest is broken into a premise and a hypothesis, and Google's T5 text-to-text transfer transformer model (9) is used to generate either "entailment" or "contradiction".

The T5 model can also be used to train question

answering (QA) tasks with a variety of formats. In a model called UnifiedQA, the T5 model is used to build a single pre-trained QA model which can handle extractive, abstractive, multiple-choice, and yes-no question answering task formats (3). This state-of-the-art UnifiedQA model can then be fine-tuned on the WinoGrande task and has been shown to achieve a 85% AUC score on the WinoGrande leaderboard.

2.3 Graph Convolutions

Graph Convolution layers (4) are often used to learn more expressive representations that help improve downstream task performance. The fact that GCN's allow linguistic representations of sentences such as parse trees to be expressed in a deep neural network is particularly useful and can be used to encode sentences meaningfully. (7) Past research shows that dependency parsing can be useful when used as input to a GCN. Zhang, et al (15) show that downstream performance on a relationship extraction task was improved by performing graph convolution over dependency trees. They hypothesize that this is due to the long-range dependencies captured by dependency parsing. Even for other coreference resolution sub-tasks, like Gendered Ambiguous Pronoun (13), we see graph-based approaches doing fairly well. Particularly, Xu and Yang (14) show that we can use GCNs to construct a more useful embedding than BERT for co-reference resolution tasks. For our purposes, we consider these graph-based approaches as strictly augmenting the CLS-based classifier approach to coreference resolution.

3 Methodology

3.1 Problem formulation

First, we formulate the problem as such: each sample generates two inputs by replacing the missing pronoun with each of the two noun options. From this, we determine how likely each resulting sentence is to be correct, telling us whether it has its pronoun replaced by the correct noun option.

Our output will be a un-normalized 1x2 float vector where the first float represents how likely the sample is to be correct and the second float represents how likely it is to be incorrect. For a given sentence, we have the following cases:

1. Only one vector has a higher likelihood for correctness
2. Both vectors have a higher likelihood for correctness
3. Both vectors have a higher likelihood for incorrectness

Source	Target
Elena liked lizards more than Rachel did, so **Rachel** was appalled to receive a pet gecko for her birthday	True
Elena liked lizards more than Rachel did, so **Elena** was appalled to receive a pet gecko for her birthday	False

Table 1: Structuring WinoGrande problems as appropriate input for BERT.

In case 1, our clear answer is the option with the higher correctness likelihood vector. In the other two cases, we take the greater absolute difference and choose our answer accordingly.

3.2 Dependency Parsing

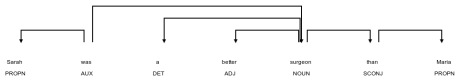

Figure 1: Dependency Parse Tree

Once we have formulated our input sentence, we use the spaCy library (2) to create the dependency trees for our input sentences. These representations capture long range dependencies between words that can often be missed by BERT-like models. We want to show that capturing this syntactic information is useful in a classification context - particularly that of co-reference resolution. When we parse the tree, we are interested in "HEAD" words and the words that modify those heads. But more specifically, we only care about the words that are related to our TARGET words (i.e. the two nouns and the pronoun). Using the dependency tree, we are able to identify which words in the sentence modify our nouns and pronoun - and give them heightened importance when we input them into our GCN layer as we'll see below.

3.3 Fine-tuning Vectors

For our model, we pretrained a Roberta (6) model using Facebook's fairseq (8) library on the Winograd Schema Challenge (WSC) dataset and on a reduced version of MASKEDWiki. The WSC was the previous benchmark for co-reference resolutions and has a less robust dataset than Wino-

Grande, and MASKEDWiki has been shown to improve performance on coreference problems when pre-trained on.

3.4 Graph Construction

Given the vectors for each token in the sentence and the dependency tree of the entire sentence, we construct the graph representation that we will provide as input to our GCN layer. We use the DGL library (12) for constructing our graph representations and performing pytorch-compatible computations on them.

We are only interested in target tokens (the nouns that are our options). So when parsing the dependency tree, we only add the nodes that are target tokens, or have target tokens as their "HEAD" nodes (i.e directly modify target nodes according to the dep. tree). We then add three types of edges - token-token, head-token and token-head edges.

Each node corresponds to a token. Each node's initial hidden state is set to the pre-trained RoBERTa embeddings for the corresponding token. This is a major part of our attempt to boost RoBERTa's performance - by building on top of the embeddings it already provides. These edges encode the syntactic information from the sentence for our GCN layer to digest and provide as input to our sequential NN.

However, since the dependency parser uses the SpaCy tokenizer, but our RoBERTa embeddings are indexed by RoBERTa's tokenizer - aligning the RoBERTa embeddings to our graph nodes was tricky. In cases where one SpaCy token corresponds to multiple BERT tokens, we average the BERT embeddings for those tokens. If multiple SpaCy tokens correspond to one BERT token, we only add the BERT embedding to the first one (since that is the one we'll be adding to the graph).

3.5 Model Architecture

Our model aims to augment the CLS token with more information about the sentence to allow our classifier to decide co-reference resolution with higher accuracy. It is inspired by Xu and Yang's approach (14) where they use a GCN to bolster BERT performance on Google's Gender Ambiguous Pronoun dataset - and even borrows some code.

The overall architecture involves a Graph Convolution Layer - which takes the graph construction from above as input and outputs one hidden state per node. We then select the hidden states from the TARGET nodes and concatenate it with the RoBERTa's CLS token for the sentence and feed it into a linear classifier, a small 2 layer sequential neural network, which decides whether the given noun substitution is correct or incorrect.

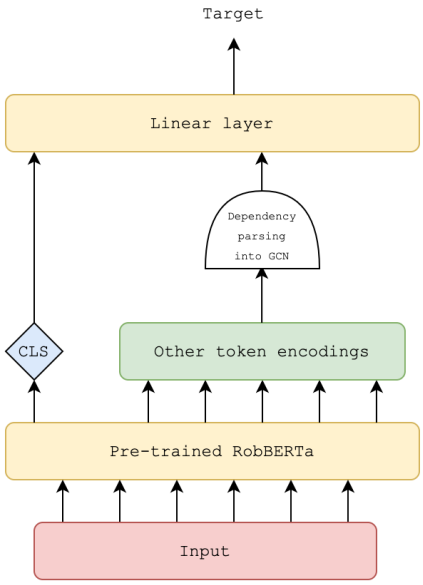

Figure 2: Model architecture

3.5.1 The GCN Layer

We use the Relational GCN proposed by Schlichtkrull et. al (11) so we can model dependencies and information flow from both head to child nodes as well as from child to head nodes (in addition to self-loops). Inspired by Xu and Yang's success, we also add in a Gate mechanism - a value between 0 and 1 - which weights the incoming graph representation. This is done to mitigate any noise from our dependency parsing library.

3.6 Regularization

We find that our model very easily overfits our training corpus, which results in good training accuracy but poor generalization to unseen data in our validation dataset. To remedy this problem, we introduce regularization in the form of adding an L2 regularization term to our loss function. This term was formulated as follows

$$Loss = Loss + \lambda \sum_{j=0}^{n} W_j^2$$

where W represents all of the weights in our model. This regularization term penalizes larger weights and thus acts to prevent our model from becoming overly complex and capturing noise or irrelevant information in our training data.

Model Size	Val Acc (No Reg)	Val Acc (Reg)
XS	0.500	0.521
S	0.470	0.530
M	0.537	0.549
L	0.539	0.563
XL	0.561	0.568

Table 2: The validation accuracies of our model, with and without regularization.

4 Results

Our model outperforms the random baseline in almost every training size, as shown in Table 3. We also outperform the CLS-based BERT baseline in many cases - which was a large part of our goal of showing that adding in syntactic information via dependency parsing is useful for improving the quality of the representation. Another curious result is that we also managed to outperform Google's T5 baseline (10).

As you can see from the table, we know that our M, L and XL models have much left to learn from their respective training sets whereas our smaller datasets have most likely overfit.

With further training and hyperparameter tuning, we hope to be able to achieve >60% accuracy when trained on the XL dataset. However, even with much room left for improvement, our model still achieves a total AUC of 0.544, which placed us in the top 20 on the Winogrande leaderboards (10).

Model Size	Train Acc	Val Acc
XS	0.938	**0.521**
S	0.857	**0.530**
M	0.808	**0.537**
L	0.798	**0.563**
XL	0.817	**0.568**
AUC	N/A	**0.544**

Table 3: The model size indicates our model architecture trained on various WinoGrande dataset sizes. Bolded results outperform the CLS-based BERT baseline.

5 Discussion

One roadblock that our model ran into during training was its inability to identify different representations of the same noun. Because our model formats the problem as generating a likelihood distribution over strictly two options, words closely related to either of the two options would not be identified as options. As an example, our model is unable to deal with problems such as the following:

> "The **man** could not dribble the **basketball** because **it** was too heavy"
> Option 1: "man"
> Option 2: "ball"

"Ball" refers to "basketball", but the GCN cannot relate them because they are not the same word.

The masked language model shines in this situation, since it masks the pronoun and predicts the masked word. This prediction is not bound by the two options, allowing the model to make such connections as the one between "ball" and "basketball", which are needed to resolve the reference.

The base CLS approach also structures itself identically to our model, and thus runs into these same aforementioned problems. However, our model performs better, which is not at all surprising given that it has the engineered features from the GCN to add representational power.

Another roadblock is the aforementioned incompatibility between Spacy tokens and BERT tokens in some instances, which led us to hacky workarounds. We would use techniques such as averaging encodings when one Spacy token was split amongst multiple BERT tokens or when two Spacy tokens were split amongst two different BERT tokens within the same word. Since we observe that the performance of our model trained on the XL dataset is only marginally better than when trained on the much smaller L dataset, we came to reason that the much higher number of faulty token discrepancies associated with the much larger XL dataset led to inaccuracy. This, we thought, was presumably enough to offset the gain in accuracy that should have resulted from multiplying the size of our training data from L to XL.

In the future, the most effective way for us to improve our validation accuracy would be to find a more effective way of resolving Spacy tokens with Bert tokens during our data preprocessing stage.

6 Further Research

Further work on this subject could involve simple improvements on our model that were infeasible with our time and computational limits.

Such improvements could include fine-tuning our RoBERTa model on the full MASKEDWiki

text corpus, as we did not have the computational resources to utilize all 130 million examples. Our current RoBERTa is fine-tuned on the WSC dataset. As we have previously mentioned, our model seemed to continue to learn even when it looked like it was overfitting to our training set. Thus, we expect that the simple augmentation of adding more training data will improve our accuracy on WinoGrande.

We also believe we have not fully explored the potential of Graph Convolutions. Further research could be done to explore different architectures of the Graph Convolutional Network, including adding additional hidden layers and making the graph more interconnected. Our current computational resources prevented us from investigating these architectures.

Our current GCN module only processes a dependency tree for a given input sentence. In the future, we could engineer additional features using other information extraction methods such as constituency parsing. We could then process our constituency tree or other tree feature with our GCN module. We believe it is likely that the CLS token fails to capture other types of tree-structured information, and thus this augmentation could further improve our results.

7 Conclusion

We observe that concatenating graph-based sentence level encodings with the CLS significantly improves its classification performance. This shows that there are some features that the CLS token actually does not encode. Specifically, since we use dependency parsing and graph representations, we hypothesize that the BERT models miss capturing long-range dependencies in their sentence representations.

We also observe that while adding this information improves the classification performance for a CLS token, it does not match performance seen by masked language modelling approaches to the task. This further shows that while the CLS token may not be able to capture sentence-level embeddings meaningfully, BERT-like models do still have unmatched language modelling capabilities which cannot be matched by custom feature engineering.

8 Final Notes

You can find our code for this paper here: https://github.com/zhang-edward/WinoGrande/

Lastly, we would like to note that all three of us are undergraduate students and this was our first deep learning project at scale. We made a wholehearted attempt at the WinoGrande challenge, and at presenting our ideas coherently in this paper. We would greatly appreciate feedback to help us improve.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018.

[2] M. Honnibal and I. Montani, "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing," 2017, to appear.

[3] D. Khashabi, T. Khot, A. Sabharwal, O. Tafjord, P. Clark, and H. Hajishirzi, "Unifiedqa: Crossing format boundaries with a single qa system," 2020.

[4] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *CoRR*, vol. abs/1609.02907, 2016. [Online]. Available: http://arxiv.org/abs/1609.02907

[5] S.-C. Lin, J.-H. Yang, R. Nogueira, M.-F. Tsai, C.-J. Wang, and J. Lin, "Tttttackling winogrande schemas," *ArXiv*, vol. abs/2003.08380, 2020.

[6] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

[7] D. Marcheggiani and I. Titov, "Encoding sentences with graph convolutional networks for semantic role labeling," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, Sept. 2017, pp. 1506–1515. [Online]. Available: https://www.aclweb.org/anthology/D17-1159

[8] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, "fairseq: A fast, extensible toolkit for sequence modeling," in *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[9] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," 2019.

[10] K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi, "Winogrande: An adversarial winograd schema challenge at scale," 2019.

[11] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," 2017.

[12] M. Wang, L. Yu, D. Zheng, Q. Gan, Y. Gai, Z. Ye, M. Li, J. Zhou, Q. Huang, C. Ma, Z. Huang, Q. Guo, H. Zhang, H. Lin, J. Zhao, J. Li, A. J. Smola, and Z. Zhang, "Deep graph library: Towards efficient and scalable deep learning on graphs," *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019. [Online]. Available: https://arxiv.org/abs/1909.01315

[13] K. Webster, M. Recasens, V. Axelrod, and J. Baldridge, "Mind the gap: A balanced corpus of gendered ambiguou," in *Transactions of the ACL*, 2018, p. to appear.

[14] Y. Xu and J. Yang, "Look again at the syntax: Relational graph convolutional network for gendered ambiguous pronoun resolution," *CoRR*, vol. abs/1905.08868, 2019. [Online]. Available: http://arxiv.org/abs/1905.08868

[15] Y. Zhang, P. Qi, and C. D. Manning, "Graph convolution over pruned dependency trees improves relation extraction," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2018. [Online]. Available: https://nlp.stanford.edu/pubs/zhang2018graph.pdf