RoBERTa Can Do More: Incorporating Syntax Into RoBERTa-based Sentiment Analysis Models Without Additional Computational Costs

Anonymous ACL submission

Abstract

We present a simple, but effective method to incorporate syntactic information obtained from dependency trees directly into transformer-based language models (e.g. RoBERTa) for tasks such as Aspect-Based Sentiment Classification (ABSC), where the desired output depends on specific input tokens. In contrast to prior approaches to ABSC that capture syntax by combining language models with graph neural networks over dependency trees, our model, Graph-integrated RoBERTa (GoBERTa) requires only a minimal increase in memory cost, training and inference time over the underlying language model. Yet, GoBERTa outperforms these more complex models, yielding new state-of-the-art results on ABSC.

1 Introduction

Aspect-Based Sentiment Classification (ABSC, Pontiki et al. (2014), Figure 1) is a fine-grained sentiment analysis task that aims to handle the fact that even simple statements such as *"The ambience was nice, but service wasn't so great."* may express different sentiments towards different aspects (this reviewer is positive about the restaurant's *"ambience"*, but negative about its *"service"*). In ABSC, the aspect to be classified is identified by a target string in the input sentence (e.g. *"ambience"*), and systems have to return the polarity (positive, neutral, negative) of the corresponding sentiment.

Pre-trained language models (PLMs) have been shown to work well for ABSC (Wang et al., 2016; Li et al., 2019; Xu et al., 2020b; Karimi et al., 2021), presumably because their attention mechanisms capture semantic connections between target and context words (Tang et al., 2016). Starting with Do et al. (2019), PLMs have been supplemented with syntactic features, typically extracted from dependency graphs. This is typically done by using the word embeddings obtained from the PLM to initialize the node embeddings of a graph neural network (GNN) obtained from the dependency graph (Wu et al., 2022; Xu et al., 2020a; Wang et al., 2020; Hou et al., 2021; Xiao et al., 2022; Tang et al., 2020; Xiao et al., 2021). However, such combined models have two major limitations:

1. **Computational Cost Problem.** Using the output embeddings of the PLM as inputs to the GNN increases both training and inference over using a PLM alone, and requires two distinct sets of parameters to be learned and stored. Since low computational demand and latency are vital for real-world applications (e.g., customer service), it is crucial to design a combination model that reduces the computational cost.

2. **Suboptimal Interaction Problem.** A typical challenge in combining PLMs and GNNs is to make the two models effectively interact with each other. Some approaches (Tang et al., 2020; Lu et al., 2020) attempt to accomplish this through heavy model architecture engineering. However, the PLM and GNN still operate in an asynchronous manner, limiting their interaction, and yielding only a minor improvement in performance. We hypothesize that more integrated models can yield larger boosts in performance.

In order to alleviate these limitations, we propose **G**raph-integrated **RoBERTa** (GoBERTa), a novel framework for effectively augmenting PLMs with syntactic information. We chose RoBERTa (Liu et al., 2019) as our PLM baseline model due to its notable performance in the ABSC task (Dai et al., 2021). GoBERTa adds three components to RoBERTa: (1) a **[g] token** that captures graph information via layer-specific attention masks, (2) a **Variable Distance Control (VDC)** hyper-parameter that defines how these attention masks depend on the graph structure, and (3) a **Variable Interaction Control (VIC)** mechanism that defines how the [g] token interacts with

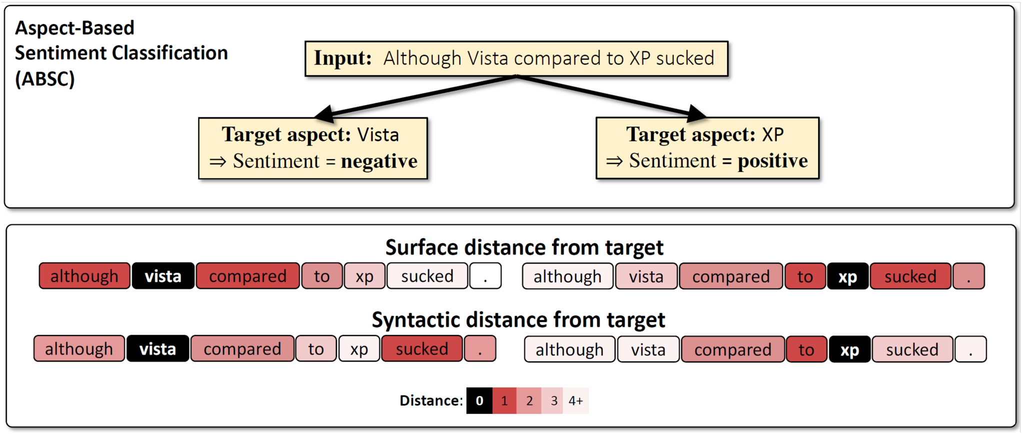


Figure 1: (Top) In ABSC, the sentiment to be predicted depends on the desired target aspect (words from the input). (Bottom) For ABSC, syntactic distance (see Fig. 2) can be more informative than surface distance.

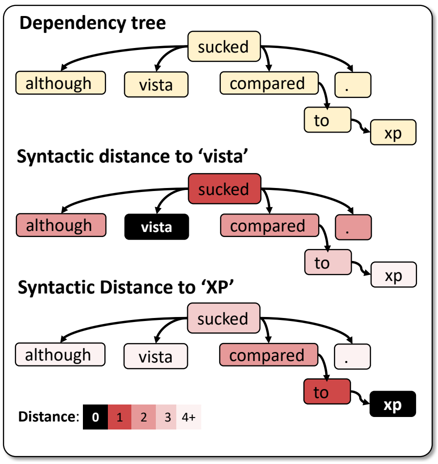


Figure 2: Dependency Trees define syntactic distances

RoBERTa's [s] token. GoBERTa outperforms prior approaches (including methods that combine PLMs and GNNs), and establishes a new state of the art, on the most widely used ABSC datasets. But since GoBERTa uses no additional parameters and its run time is almost identical to RoBERTa itself ($< 0.5\%$ increase), it solves the computational cost problem.

2 Aspect-Based Sentiment Classification

In Aspect-Based Sentiment Classification (Pontiki et al., 2014), illustrated in Figure 1, the task is to predict the polarity (positive, negative or neutral) of the sentiment in input sentence $s = [w_1, w_2, ..., w_p, ..., w_{p+m-1}, ..., w_n]$ towards a given target aspect t (a substring of the input sentence: $t_i = \{w_p, ..., w_{p+m-1}\}$).

2.1 Language Models for ABSC

Large pre-trained language models (PLMs) such as BERT (Devlin et al., 2019), XLNET (Yang et al., 2019), and RoBERTa (Liu et al., 2019) have gained predominance for many NLP tasks, including ABSC. RoBERTa, a variant of BERT, is known to show notable performance on ABSC tasks (Dai et al., 2021), and forms the basis of the models explored in this paper. RoBERTa (and BERT) are (pre)trained on large amounts of raw text with a masked language modeling objective. Both models use a Transformer (Vaswani et al., 2017) architecture in which each token's embedding is fed through multiple layers such that each token's embedding in a given layer can attend to all tokens in the sequence (in the same layer). To adapt these models for classification tasks, a special token ([CLS] for BERT, [s] for RoBERTa) whose output is fed into a task-specific feedforward layer is included in the input sequence. A separation token ([SEP] or [/s]) can be used to separate the input sequence from other task-specific information.

For the ABSC task, RoBERTa is typically used as follows: after tokenization, the input sentence is fed into RoBERTa as '[s] input sentence [/s] [/s] aspect sequence [/s]', where the aspect sequence includes the target aspect word itself. Only the [s] token embedding of the last layer is used for the final prediction and fine-tuning.

2.2 Combining PLMs with syntax

A common approach to ABSC is to supplement a PLM with syntactic information (Tang et al., 2020; Zhang et al., 2019b) obtained from a dependency parser. In a dependency graph (Figure 2) each word in the sentence corresponds to a node, with labeled edges indicating word-word dependencies. Note that the syntactic distance between related words (e.g. *sucked* and *vista*) can be much smaller than their surface distance in the original sequence.

Since the dependency parser and the PLM may use different tokenizers, tokenization needs to be broken into two stages to integrate both models seamlessly. The input sentence is first tokenized by the dependency parser, and then each token is again tokenized by RoBERTa's tokenizer, following previous work (Tang et al., 2020).

Graph Neural Network-based ABSC models To incorporate syntax into ABSC models, PLMs have been augmented with Graph Neural Networks (GNNs, Kipf and Welling (2016)) that capture the structure of the sentence's dependency tree. Although there are many variants (Trisna and Jie, 2022), the basic idea behind GNNs is to represent each node as a vector h_i that is updated via graph convolution in each layer ($l \in [1, 2, \ldots L]$) of the GNN (Kipf and Welling, 2016) by aggregating its neighborhood information from the previous layer:

$$h_i^l = \sigma(A_{ij} W_l h_j^{l-1} + b_l h_j^{l-1})$$

Here σ is is an activation function, W and b are learnable parameters, and A_{ij} is the entry of the graphs adjacency matrix that indicates whether nodes i and j are connected (in which case $A_{ij} = 1$; otherwise $A_{ij} = 0$). If A is defined by a dependency tree, $A_{ij} = 1$ if there is a dependency between words i and j. To combine GNNs with PLMs for ABSC, the GNN embeddings of all words can be initialized with the PLM's output embeddings, and the embeddings of the target aspects in the last layer can be used for classification. Zhang et al. (2019a) was the first to implement a GNN-based model for ABSC, adding a multi-layered Graph Convolutional Network (GCN) to encode dependency graphs on top of the word embedding layer. Sentic GCNs (Liang et al., 2022) leverage the dependencies between context words and aspect words on top of the embedding module. Wang et al. (2020) and Wu et al. (2022) used a relational graph attention network (R-GAT) on top of initial embeddings from BERT. Tang et al. (2020) presented a dependency graph enhanced dual-transformer network named DGEDT that contextual representation and graph representation interact with each other through a mutual biaffine module. More recent research in ABSC has tried to revise dependency graphs due to the noise and imperfection of syntactic dependency graphs (Xiao et al., 2021, 2022). What is common to all these approaches is that the PLM and GNN operate in a serial fashion, and are not tightly integrated.

Attention-mask based approaches Another promising approach to incorporate syntactic information into PLMs that is more related to this paper, is to manipulate the Transformer's self-attention masks. For example, Syntax-BERT (Bai et al., 2021) uses multiple masks induced from the syntactic trees (e.g., parent, children, sibling, pairwise masks) to incorporate syntactic information into BERT. To do so, it requires multiple (usually more than 90) sub-networks, which causes a considerable amount of increase in training/inference time. The key difference between Syntax-BERT and GoBERTa is that Syntax-BERT alters all the input tokens' attention masks while GoBERTa (which is specifically designed for tasks like ABSC, where the desired output depends on specific parts of the input) keeps the original input tokens intact while only modifying the attention-mask of the newly added [g] token. This allows GoBERTa to meet our primary objective of keeping the computational costs constant.

3 GoBERTa

The primary objective of GoBERTa (Figure 4) is to incorporate syntactic information into a PLM without (essentially) increasing the computational costs (i.e. number of model parameters and running time) of the PLM. We accomplish this goal by augmenting RoBERTa with three components: (1) a single additional input token, named [g], whose attention masks depend on the structure of the input's dependency tree(s), paired with (2) a "variable distance control" (VDC) mechanism that specifies how the structure of the dependency graph is reflected in [g]'s attention masks, and additionally (3) a "variable interaction control" (VIC) mechanism that specifies the interaction of [g] and [s]. Since GoBERTa does not introduce any new learnable parameters and only increases the sequence length of every input by one token ([g]), it has nearly identical training/inference time (less than 1 % increase) to the standard RoBERTa model.

Input and output After tokenization with RoBERTa's tokenizer, the input to GoBERTa is '[s] [g] input sentence [/s] [/s] aspect sequence [/s]',

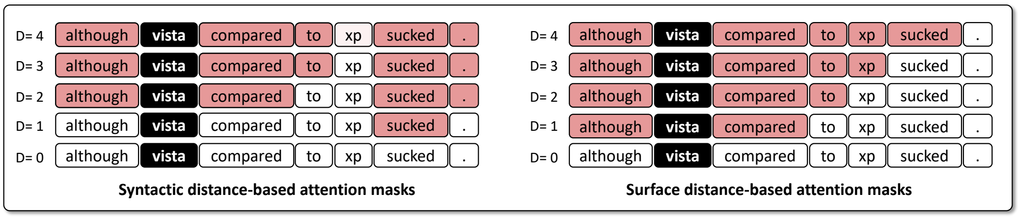


Figure 3: GoBERTa's [g] token uses attention masks based on syntactic distance (left), not surface distance (right)

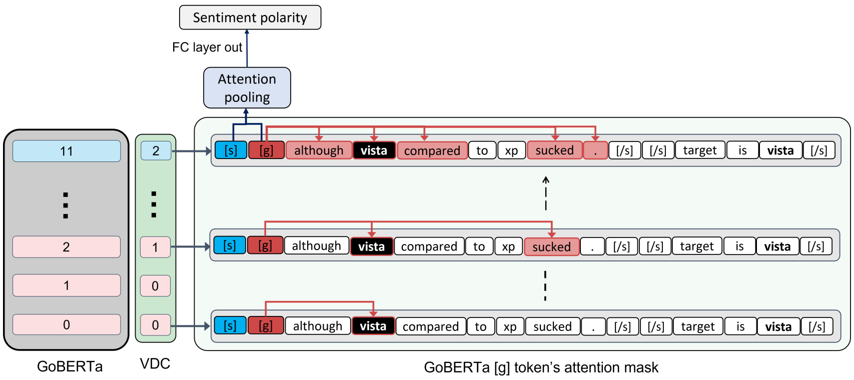


Figure 4: The overall architecture of GoBERTa with Variable Distance Control.

where the aspect sequence is the phrase "target is" followed by the target aspect words (this gave slightly better performance than using only the aspect words). [g] and [s] use the same dictionary embedding in the input layer. We evaluate this choice in Section 5.[1] To obtain the output, the final layers of the [s] and [g] tokens are pooled before feeding them through a softmax classification layer. For the pooling process, we use the attention-based pooling mechanism introduced in (Bai et al., 2019).

The [g] token and distance-based attention masks To capture the intuition that the relevance of each word in a sentence to ABSC depends on its distance to the target aspect words, we define distance-based attention masks (Figure 3) that depend either on syntactic or surface distance, and are only used for the [g] token. For a given distance metric $D(j)$ and distance d_i, an attention mask $\mathbf{m}_i$ is a vector whose elements $\mathbf{m}_{ij}$ are zero if the distance $D(j)$ between token j and the target aspect words is greater than d_i, and one otherwise. If distance is syntax-based, $D(j)$ is the length of the shortest path between token j and the target aspect (so, if the target aspect consists of multiple tokens, we take the minimum distance to any of its component tokens). If the distance is surface-based, $D(j)$ is simply the token distance to the target aspect (1 if j is adjacent). The [g] token is inserted next to the [s] token. Unlike the [s] token that attends to every token in the input, each layer l_i of [g] only attends to the subset of input tokens that are at most a distance d_i (specified by the VDC hyperparameters explained below) away from the target aspect. We do not restrict how the input tokens can attend to [g]. The attention between [s] and [g] is controlled by the VIC hyperparameters described below.

Variable Distance Control (VDC) To specify the attention masks used by the [g] token, we introduce a new set of hyper-parameters named Variable Distance Control (VDC). VDC is a list of 12 non-negative integers where the i-th element represents the value of d_i of the i-th layer of the [g] token. For example, if the VDC is $[0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1]$, the first six layers of the [g] token attend only to the target aspect, and the remaining six layers attend to tokens that are connected to the target via a direct dependency link.

Note that increasing VDCs (e.g., [0,0,0,0,1,1,1,1,2,2,2,2], [0,0,0,1,1,1,2,2,2,3,3,3]) can be used to mimic how GNNs' work. Through graph convolution, the i-th layer of a GNN aggregates features of nodes up to length i away from each node in the graph, allowing the GNN to gradually aggregate information from more and more distant nodes in its upper layers. Empirical

[1]Future work could examine if letting [g]'s embedding vary independently of [s]'s during fine-tuning would be beneficial.

results in Section 5 show that increasing VDCs have indeed better performance than constant VDCs (e.g., [2,2,2,2,2,2,2,2,2,2,2,2]) or decreasing VDCs (e.g. [3,3,3,2,2,2,1,1,1,0,0,0]).

Variable Interaction Control Unlike [g], the [s] token always attends to the entire input sequence. To make the best of use of both types of information, the interaction between them is crucial (Tang et al., 2020). Unlike previous combination models where syntax is captured by a distinct model, GoBERTa integrates it directly into the PLM, and since it does so by adding a separate [g] token, we can also use an attention mask mechanism to precisely control the interaction between [s] and [g] in each layer. For example, we can allow [s] and [g] to attend to themselves and each other ("full interaction"), only to themselves ("self interaction"), or only to each other ("cross interaction"), as in Figure 5. GoBERTa has an additional set of hyperparameters, called variable interaction control (VIC), that define how [s] and [g] interact in each layer. Although there are theoretically 16 possible VIC values for each of the 12 layers, we only experiment with the three settings shown in Figure 5. We show in Section 5 that starting with n self interaction layers as a warm-up phase and then transitioning to $(12 - n)$ cross interaction layers can boost the performance of GoBERTa.

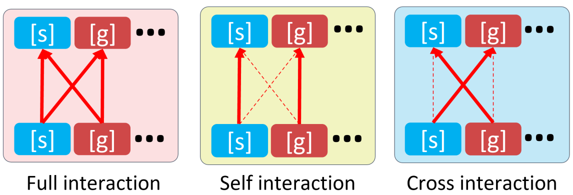


Figure 5: The Variable Interaction Control (VIC) hyperparameters define how [s] and [g] attend to each other and themselves. We experiment with the three of the 16 possible VIC values shown here ("full", "self" and "cross" interaction)

4 Experimental Results

Datasets and Experimental Settings We use the most widely used ABSC data sets: the Laptop and Restaurant datasets from SemEval-2014 task 4 (Pontiki et al., 2014) and the Twitter dataset of Dong et al. (2014). Table 7 in Appendix A shows the statistics of the ABSC datasets. For GoBERTa, we use the pre-trained RoBERTa-base model[2] provided by huggingface. We use spaCy[3]'s en_core_web_sm model version 3.3.0 as dependency parser. Finetuning uses a batch size of 32, dropout rate of 0.1, and learning rate of 1.5e-5 using the AdamW optimizer. We run the experiments with five random seeds and report the average accuracy and macro-F1. All the experiments are conducted on a single Tesla A100 GPU.

Overall Results Table 1 compares GoBERTa against all competitive RoBERTa+GNN or BERT+GNN combination models that use dependency graphs extracted from widely used dependency tree parser such as spaCy[3], Stanford CoreNLP[4], and Biaffine Parser[5]. We can see that GoBERTa outperforms all previous models on both SemEval-2014 Task4 datasets, establishing a new state-of-the-art record. On Twitter, GoBERTa clearly outperforms the other RoBERTa based models and is competitive with the (overall better performing) BERT-based models. However, since [g] uses the same parameters GoBERTa has the exact same number of parameters as the basic RoBERTa model, and only requires minute ($\leq 0.5\%$) increases in training and inference run times (Table 2), it arguably resolves the computational cost problem mentioned in Section 1.

Twitter and multi-sentence items Table 3 shows that the Twitter dataset has a particularly large proportion of multi-sentence items. Since each sentence has a single dependency graph, multi-sentence items have multiple dependency graphs, requiring us to combine them by adding a dummy root node that links to the heads of each sentence. This, as well as RoBERTA's generally lower performance on Twitter, may be one reason why we do not achieve state of the art on Twitter. We have also not attempted to examine how parser accuracy contributes to performance differences across datasets

5 Analysis

We now examine the effect of the design decisions and hyperparameters that distinguish GoBERTa from RoBERTa through a number of analyses and ablation studies.

[2] https://huggingface.co/roberta-base
[3] https://spacy.io/
[4] https://stanfordnlp.github.io/CoreNLP/
[5] Biaffine Parser (Dozat and Manning, 2016) implemented from the allenNLP https://allenai.org/allennlp

Base PLM	Models	Lap14		Rest14		Twitter	
		Acc.	F1	Acc.	F1	Acc.	F1
BERT	DGEDT-BERT[3](Tang et al., 2020)	79.8	75.6	86.3	80.0	77.9	75.4
	RGAT-BERT[5](Wang et al., 2020)	78.2	74.1	86.6	81.4	76.2	74.9
	DGNN (BERT)[5] (Xiao et al., 2022)	81.4	79.0	87.2	81.7	76.2	75.0
	PD-RGAT (BERT)[4] (Wu et al., 2022)	81.6	80.9	88.7	83.6	77.9	76.2
	MWM-GCN (BERT)[4] (Zhao et al., 2022)	82.8	79.5	88.5	82.6	**78.9**	**77.4**
	Sentic GCN-BERT[3] (Liang et al., 2022)	82.1	79.1	86.9	81.0	–	–
	SGGCN-BERT (Veyseh et al., 2020)	82.8	80.2	87.2	82.5	–	–
RoBERTa	BERT4GCN (RoBERTa)[3] (Xiao et al., 2021)	81.8	78.2	86.2	78.6	74.8	74.0
	RoBERTa-RGAT[5](Dai et al., 2021)	83.4	80.3	87.4	80.6	74.4	72.9
	RoBERTa-PWCN[3] (Dai et al., 2021)	84.2	81.2	87.4	81.1	76.6	75.6
	Ours: GoBERTa [3]	**84.5**	**81.6**	**89.3**	**84.3**	77.2	76.0

Table 1: GoBERTa outperforms all prior works on the Laptop and Restaurant data, and is competitive on Twitter

Model	# of Params.	Training (s)	Inference (s)
RoBERTa	125M	12.77	0.3452
GoBERTa	125M (+0.0%)	12.84 (+0.5%)	0.3464 (+0.3%)

Table 2: Computational cost comparison between GoBERTa and a single RoBERTa. RoBERTa and GoBERTa has the exact same number of total parameters. The reported run times are measured as the average of 100 runs for a single epoch in the Twitter dataset. We use batch size of 32 and a single Tesla A100 GPU.

Distribution	Datasets		
		Train	Test
Lap14	% of multiple sent./item	7.86	7.84
	Avg. sent./item	1.09	1.09
Res14	% of multiple sent./item	4.02	4.38
	Avg. number of sent./item	1.04	1.05
Twitter	% of multiple sent./item	59.44	60.55
	Avg. number of sent./item	1.99	1.96

Table 3: Prevalence of multi-sentence items in the ABSC datasets.

Does [g] require syntactic distances? To understand the impact of syntax on GoBERTa, we now compare it to a variant that uses surface distance instead of syntactic distance. The surface (or position) distance of a token is computed simply by the number of tokens between the closest target aspect token and the corresponding token following previous works (Zeng et al., 2019; Phan and Ogunbona, 2020). Focusing on words near the target aspect is known to be effective in the ABSC task (Zeng et al., 2019). But syntactic distance is often very different from surface distance (see Figures 1 and 3, where the target word 'vista' and the sentiment word 'sucked' are not connected until $D = 4$ when using the position distance, while the dependency graph captures the connection between 'vista' and 'sucked' at $D = 1$). In fact, Dai et al. (2021) have observed that the average syntactic distances (based on dependency graphs) between target and sentiment words are 3.77 and 4.46 for the laptop and restaurant datasets, while the average position distances are 6.48 and 7.49 respectively.

Table 4 shows results for all three VDCs types (decreasing, constant, and increasing) under both metrics that indicate that syntactic distances yield generally better performance than position-based distances, especially in the increasing VDC configuration.

The Impact of Variable Distance Control GoBERTa is inspired by how GNNs aggregate information from nodes that are more and more distant in their upper layers. As mentioned in section 3, increasing VDC hyperparameters can be used to mimic this behavior. As mentioned above, Table 4 summarizes experiments conducted on three different types of VDCs: increasing (e.g., [0,0,0,1,1,1,2,2,2,3,3,3]), constant (e.g., [2,2,2,2,2,2,2,2,2,2,2,2]), and decreasing (e.g., [3,3,3,2,2,2,1,1,1,0,0,0]). It can be seen that GoBERTa has the highest performance with increasing VDCs (i.e. when it is most similar to typical GNNs), and the lowest performance with decreasing VDCs (i.e. when it is the least similar to GNNs). More detailed experiment results are provided in Appendix C.

What range of distances matters for ABSC? Finally, Dai et al. (2021)'s observation that different corpora exhibit different distances and that syntactic distances are shorter than surface distances is also consistent with the results in Figure 6. Here, we use a constant VDC, but vary its range from

Variable Distance Control (VDC)	Lap14		Rest14		Twitter	
	Acc.	F1	Acc.	F1	Acc.	F1
RoBERTa-ASC	82.1	78.9	87.6	81.7	75.6	74.5
GoBERTA (Position Distance)						
• Decreasing-VDC	83.4	80.4	88.5	83.2	76.5	75.3
• Constant-VDC	83.7	80.7	88.6	83.3	76.4	75.4
• Increasing-VDC	83.7	80.5	88.6	83.2	76.9	**76.0**
GoBERTA (Dependency Graph)						
• Decreasing-VDC	83.7	80.6	88.4	83.0	76.5	75.4
• Constant-VDC	83.7	80.4	88.9	83.7	76.4	75.2
• Increasing-VDC	**83.8**	**80.8**	**89.1**	**83.8**	**77.1**	75.9

Table 4: Empirical results on the effect of VDC. The results show that GoBERTa generally shows better performance in the order of decreasing $<$ fixed $<$ increasing VDCs. This result matches our intuition of [g] imitating GNN as described in Section 3. A more detailed result table is in the Appendix C.

Variable Interaction Control (VIC)	Lap14		Rest14		Twitter	
	Acc.	F1	Acc.	F1	Acc.	F1
GoBERTA w/o Variable Interaction	83.8	80.8	89.1	83.8	77.1	75.8
GoBERTA w/ Variable Interaction						
• Cross $(n) \rightarrow$ Self $(12-n)$						
• $n=4$	83.3	80.4	88.7	83.5	75.6	74.3
• $n=6$	83.5	80.3	88.4	82.9	74.8	73.3
• $n=8$	82.7	79.3	88.3	82.8	76.0	74.7
• Self $(n) \rightarrow$ Cross $(12-n)$						
• $n=4$	84.2	80.9	**89.3**	**84.3**	75.7	74.6
• $n=6$	84.1	81.0	89.0	83.8	**77.2**	**76.0**
• $n=8$	**84.5**	**81.6**	89.1	84.1	76.9	**76.0**

Table 5: Empirical results on the effectiveness of VIC. See Figure 5 for the definitions of self and cross interactions. We use increasing VDCs [000011112222] for Laptop and Twitter and [000222444666] for Restaurant.

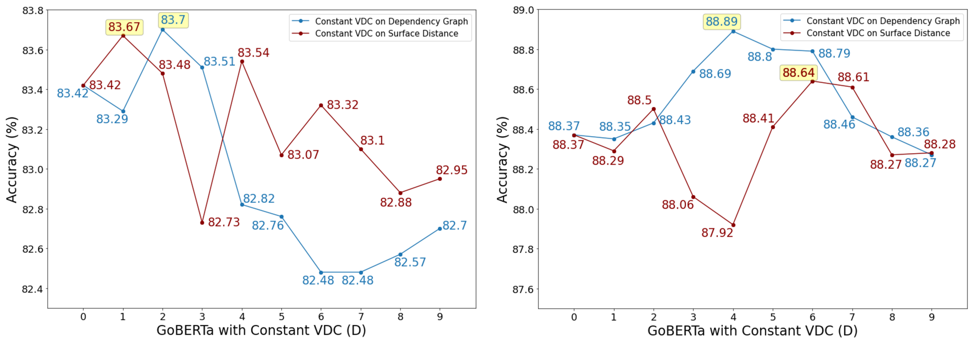
0 to 9 across runs. Using surface distance (red dot in Figure6), performance peaks near D=1–4 on the laptop data, and near D=6,7 on the restaurant dataset. On the other hand, when using syntax distances (blue dot in Figure6), performance peaks near D=2 for the laptop data, and near D=4,6 on the restaurant data.

The Impact of Variable Interaction Control As explained in Section 3, the VIC hyper-parameters allow us to control the degree of interaction between the [s] and [g] token in each layers.

Although there are 16^{12} possible VIC configurations (4 options per [s] and [g] token, in each of the 12 layers), we only experiment with the three VIC settings shown in Figure 5, and only explore a full variant (where all layers use full interactions), one variant where GoBERTa first goes through n self-interaction layers and then transitions to $(12-n)$ cross-interaction layers, and a reverse ordering where cross-interaction happens in the first n layers, followed by $(12-n)$ self-interaction layers. The results for $n \in 4, 6, 8$ are summarized in Table 5. Starting with n self interaction layers and then transitioning to $(12-n)$ cross interaction layers generally outperforms using only constant interaction. On the other hand, going through cross interaction layers first and then through self interactions generally shows worse performance.

Although we have only examined a small number of possible VIC configurations, we can see that the VIC settings can have a significant impact on performance. Finding the best VIC configuration (or combination of VIC and VDC configurations) could be an interesting future work.

Does [g] need to be a separate token? We now compare GoBERTa to a variant that does not use a [g] token, but instead uses the target tokens at the end of the input sequence (recall that the input sequence has the form of '[s] sentence [/s] [/s] target is aspect [/s]'). We call this the GoBERTa-[g] variant. As Table 6 shows, the loss in performance is considerable compared to using an independent [g] token as in the the original GoBERTa model. We speculate that the drop in performance is due to the original input sentence getting corrupted when



(a) Laptop Dataset

(b) Restaurant Dataset

Figure 6: **Experiments on different constant VDC values** This result implies that the restaurant data has a longer distance between sentiment word and target than the laptop data.

[g] token	Lap14		Rest14		Twitter	
	Acc.	F1	Acc.	F1	Acc.	F1
GoBERTa-[g]	83.5	80.5	88.3	82.9	75.6	74.3
GoBERTa						
[g] init. = [s] embed.	**83.8**	**80.8**	**89.1**	**83.8**	**76.7**	**75.5**
[g] init. = aspect embed.	83.5	80.6	88.8	83.3	73.9	72.8

Table 6: Empirical results on the necessity of the [g] token and the inherent strength of the pre-trained [s] token embedding. We used the increasing VDC ([0,0,0,1,1,1,2,2,2,3,3,3]) with default VIC for the ablation studies.

we modify the aspect token's attention mask. This result indicates the importance of using an additional and independent [g] token for the GNN role as in GoBERTa.

Furthermore, there seems to be an inherent advantage in using the pre-trained embedding of the [s] token also for [g]. Table 6 also compares GoBERTa (in which the dictionary embedding of [g] is identical to [s]), with a variant in which we use the actual aspect word's dictionary embeddings as the dictionary [g] embedding (if the aspect consists of several words, we average their embeddings). Initializing [g] token with the [s] token embedding yields better performance, perhaps because the [s] embedding is better suited to aggregate information than the embeddings of other tokens, providing a better starting point for a sequence element that is also intended to capture aggregate information (albeit of a slightly different nature). We plan to examine the effect of letting [g]'s embedding deviate from [s] during fine-tuning.

6 Conclusion

This paper has proposed a novel framework, GoBERTa, that effectively incorporates syntactic information directly into a pre-trained large language model (PLM) such as RoBERTa for tasks like Aspect-Based Sentiment Classification (ABSC), in which the desired output depends on specific words in the input, and where syntactic distance to the relevant input words may be important. In contrast to prior work, where a separate GNN was added to the output of the PLM, in our model, attention masks for new [g] token capture syntactic information, and a new hyper-parameter, named variable distance control (VDC), can instead capture graph structure in a similar fashion. Another unique hyper-parameter called variable interaction control (VIC) increases the flexibility of our model by making it possible to adjust the degree of interaction between syntax and the PLM. To the best of our knowledge, GoBERTa is the first model to incorporate syntactic knowledge into RoBERTa without (essentially) increasing the computational costs. Experimental results show that we achieve state-of-the-art performance in SemEval-2014 task 4 with computational costs that are equivalent to a basic RoBERTa model. This demonstrates the efficiency of our approach and suggests a new paradigm for combining PLM and syntactic information in ABSC, even though GoBERTa is a very simple extension to RoBERTa. In future work, we plan to incorporate edge-type and/or edge-direction information into GoBERTa, and to explore the space of possible VDC and VIC settings in a more systematic fashion.

References

Jiangang Bai, Yujing Wang, Yiren Chen, Yaming Yang, Jing Bai, Jing Yu, and Yunhai Tong. 2021. Syntax-BERT: Improving pre-trained transformers with syntax trees. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3011–3020, Online. Association for Computational Linguistics.

Yunsheng Bai, Hao Ding, Song Bian, Ting Chen, Yizhou Sun, and Wei Wang. 2019. Simgnn: A neural network approach to fast graph similarity computation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 384–392.

Junqi Dai, Hang Yan, Tianxiang Sun, Pengfei Liu, and Xipeng Qiu. 2021. Does syntax matter? a strong baseline for aspect-based sentiment analysis with RoBERTa. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1816–1829, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Hai Ha Do, PWC Prasad, Angelika Maag, and Abeer Alsadoon. 2019. Deep learning for aspect-based sentiment analysis: A comparative review. *Expert Systems with Applications*, 118:272–299.

Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive recursive neural network for target-dependent Twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 49–54, Baltimore, Maryland. Association for Computational Linguistics.

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

Xiaochen Hou, Peng Qi, Guangtao Wang, Rex Ying, Jing Huang, Xiaodong He, and Bowen Zhou. 2021. Graph ensemble learning over multiple dependency trees for aspect-level sentiment classification. *arXiv preprint arXiv:2103.11794*.

Akbar Karimi, Leonardo Rossi, and Andrea Prati. 2021. Adversarial training for aspect-based sentiment analysis with bert. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 8797–8803.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Xin Li, Lidong Bing, Wenxuan Zhang, and Wai Lam. 2019. Exploiting bert for end-to-end aspect-based sentiment analysis. *arXiv preprint arXiv:1910.00883*.

Bin Liang, Hang Su, Lin Gui, Erik Cambria, and Ruifeng Xu. 2022. Aspect-based sentiment analysis via affective knowledge enhanced graph convolutional networks. *Knowledge-Based Systems*, 235:107643.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zhibin Lu, Pan Du, and Jian-Yun Nie. 2020. Vgcn-bert: augmenting bert with graph embedding for text classification. *Advances in Information Retrieval*, 12035:369.

Minh Hieu Phan and Philip O Ogunbona. 2020. Modelling context and syntactical features for aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3211–3220.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2016. Effective LSTMs for target-dependent sentiment classification. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3298–3307, Osaka, Japan. The COLING 2016 Organizing Committee.

Hao Tang, Donghong Ji, Chenliang Li, and Qiji Zhou. 2020. Dependency graph enhanced dual-transformer structure for aspect-based sentiment classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6578–6588, Online. Association for Computational Linguistics.

Komang Wahyu Trisna and Huang Jin Jie. 2022. Deep learning approach for aspect-based sentiment classification: A comparative review. *Applied Artificial Intelligence*, pages 1–37.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *ArXiv*, abs/1706.03762.

Amir Pouran Ben Veyseh, Nasim Nour, Franck Dernoncourt, Quan Hung Tran, Dejing Dou, and Thien Huu Nguyen. 2020. Improving aspect-based sentiment analysis with gated graph convolutional networks and syntax-based regulation. *arXiv preprint arXiv:2010.13389*.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020. Relational graph attention network for aspect-based sentiment analysis.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615, Austin, Texas. Association for Computational Linguistics.

Haiyan Wu, Zhiqiang Zhang, Shaoyun Shi, Qingfeng Wu, and Haiyu Song. 2022. Phrase dependency relational graph attention network for aspect-based sentiment analysis. *Knowledge-Based Systems*, 236:107736.

Luwei Xiao, Yun Xue, Hua Wang, Xiaohui Hu, Donghong Gu, and Yongsheng Zhu. 2022. Exploring fine-grained syntactic information for aspect-based sentiment classification with dual graph neural networks. *Neurocomputing*, 471:48–59.

Zeguan Xiao, Jiarun Wu, Qingliang Chen, and Congjian Deng. 2021. BERT4GCN: Using BERT intermediate layers to augment GCN for aspect-based sentiment classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9193–9200, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kuanhong Xu, Hui Zhao, and Tianwen Liu. 2020a. Aspect-specific heterogeneous graph convolutional network for aspect-based sentiment classification. *IEEE Access*, 8:139346–139355.

Qiannan Xu, Li Zhu, Tao Dai, and Chengbing Yan. 2020b. Aspect-based sentiment classification with multi-attention network. *Neurocomputing*, 388:135–143.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Biqing Zeng, Heng Yang, Ruyang Xu, Wu Zhou, and Xuli Han. 2019. Lcf: A local context focus mechanism for aspect-based sentiment classification. *Applied Sciences*, 9(16):3389.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019a. Aspect-based sentiment classification with aspect-specific graph convolutional networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4568–4578, Hong Kong, China. Association for Computational Linguistics.

Chen Zhang, Qiuchi Li, and Dawei Song. 2019b. Syntax-aware aspect-level sentiment classification with proximity-weighted convolution network. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1145–1148.

Ziguo Zhao, Mingwei Tang, Wei Tang, Chunhao Wang, and Xiaoliang Chen. 2022. Graph convolutional network with multiple weight mechanisms for aspect-based sentiment analysis. *Neurocomputing*, 500:124–134.

A Details on Datasets

Our model GoBERTa is evaluated on three different datasets from SemEval 2014 Task 4 and Twitter datasets. Table 7 shows the statistics of the datasets.

Dataset	Train	Test
Restaurant (SemEval-2014)	3608	1120
Laptop (SemEval-2014)	2328	638
Twitter	6248	692

Table 7: Dataset Overview

B Comparing Different Pooler Types

The [s] and [g] token outputs are combined after the last layer of GoBERTa encoders as described in Section 3. We conduct experiments on three different types of poolers for combining [s] and [g] token embeddings at the final layer: average, max, and attention pooling. Table 8 summarizes the results of using different pooler types for GoBERTa. The result shows that attention pooling shows better results in general.

Pooler types	Lap14		Rest14		Twitter	
	Acc.	F1	Acc.	F1	Acc.	F1
GoBERTa						
w/ max pooling	83.2	80.0	88.7	83.3	74.8	73.7
w/ avg pooling	**83.8**	80.6	88.8	83.5	76.5	**75.5**
w/ att pooling	**83.8**	**80.8**	**89.1**	**83.8**	**76.7**	**75.5**

Table 8: Comparing different pooler types for GoBERTa. We used VDC = [0,0,0,1,1,1,2,2,2,3,3,3] with the default full-interaction for the experiment.

C Detailed Variable Distance Control Results

Our variable distance control (VDC) is a unique hyper-parameter which consists of 12 non-negative integers, where each integer represents the d_i value of the i-th layer. Theoretically there are exponentially many possible values for VDC but we use three representative types: increasing, constant, and decreasing VDCs.

We heuristically chose specific values for each type of VDCs and the detailed results are summarized in Table 9. The table shows that GoBERTa has the highest performance with increasing VDCs. Increasing VDCs are designed to work as the most similar to the typical GNN by aggregating information from the closest nodes to farther nodes based on the target aspect. On the other hand, decreasing VDCs has the lowest performance due to the fact that the decreasing VDCs are designed to work as least similar to a GNN in the opposite order (i.e., aggregating information from farther nodes to closer nodes based on the target aspect). From these results, we can conclude that GoBERTa successfully imitates the typical GNN mechanism through increasing VDC configuration.

Variable Distance Control (VDC)	Lap14		Rest14		Twitter	
	Acc.	F1	Acc.	F1	Acc.	F1
GoBERTA (Position Distance)						
• *Decreasing-VDC*	83.4	80.4	88.5	83.2	76.5	75.3
• VDC = [222211110000]	83.3	80.2	88.2	82.7	76.0	74.8
• VDC = [333222111000]	82.0	78.8	88.4	82.8	**76.5**	**75.3**
• VDC = [444422220000]	**83.4**	**80.4**	88.4	83.1	75.6	74.4
• VDC = [554433221100]	83.1	79.9	**88.5**	**83.2**	75.2	73.7
• VDC = [666444222000]	83.2	80.0	87.8	82.0	76.0	74.8
• *Constant-VDC*	83.7	80.7	88.6	83.3	76.4	75.4
• Please refer to Figure 6						
• *Increasing-VDC*	83.7	80.5	88.6	83.2	76.9	76.0
• VDC = [000011112222]	83.5	80.3	87.8	82.1	**76.9**	**76.0**
• VDC = [000111222333]	83.5	**80.5**	88.5	**83.2**	75.5	74.4
• VDC = [000022224444]	83.6	80.4	87.9	82.2	75.7	74.4
• VDC = [001122334455]	83.3	80.3	**88.6**	83.1	76.1	74.9
• VDC = [000222444666]	**83.7**	**80.5**	88.3	82.5	76.6	75.8
GoBERTA (Dependency Graph)						
• *Decreasing-VDC*	83.7	80.6	88.4	83.0	76.5	75.4
• VDC = [222211110000]	83.5	80.4	88.1	82.7	75.4	74.2
• VDC = [333222111000]	82.6	79.6	87.1	81.1	76.1	75.1
• VDC = [444422220000]	83.4	80.5	87.9	82.2	**76.5**	**75.4**
• VDC = [554433221100]	**83.7**	**80.6**	**88.4**	**83.0**	75.3	74.2
• VDC = [666444222000]	83.2	80.0	88.2	82.7	75.6	74.3
• *Constant-VDC*	83.7	80.4	88.9	83.7	76.4	75.2
• Please refer to Figure 6						
• *Increasing-VDC*	83.8	80.8	89.1	83.8	77.1	75.9
• VDC = [000011112222]	83.5	80.5	88.3	82.8	**77.1**	75.8
• VDC = [000111222333]	**83.8**	**80.8**	**89.1**	**83.8**	76.7	75.5
• VDC = [000022224444]	83.5	80.5	88.9	83.5	75.7	74.6
• VDC = [001122334455]	83.2	80.2	88.8	83.5	74.7	75.9
• VDC = [000222444666]	82.5	79.4	88.9	**83.8**	76.9	**75.9**

Table 9: Detailed experimental results on the effect of DRC. The results show that GoBERTa generally shows better performance in the order of decreasing $<$ fixed $<$ increasing DRCs. This result matches our intuition of [g] token imitating GNN as described in Section 3.