GiBERT: Enhancing BERT with Linguistic Information using a Lightweight Gated Injection Method

Nicole Peinelt[1,2] and **Marek Rei**[3] and **Maria Liakata**[1,2,4]
[1]The Alan Turing Institute, UK
[2]University of Warwick, UK
[3]Imperial College London, UK
[4]Queen Mary University of London, UK
n.peinelt@warwick.ac.uk, marek.rei@imperial.ac.uk,
m.liakata@qmul.ac.uk

Abstract

Large pre-trained language models such as BERT have been the driving force behind recent improvements across many NLP tasks. However, BERT is only trained to predict missing words – either through masking or next sentence prediction – and has no knowledge of lexical, syntactic or semantic information beyond what it picks up through unsupervised pre-training. We propose a novel method to explicitly inject linguistic information in the form of word embeddings into any layer of a pre-trained BERT. When injecting counter-fitted and dependency-based embeddings, the performance improvements on multiple semantic similarity datasets indicate that such information is beneficial and currently missing from the original model. Our qualitative analysis shows that counter-fitted embedding injection is particularly beneficial, with notable improvements on examples that require synonym resolution.

1 Introduction

Detecting the semantic similarity between a given text pair is at the core of many NLP tasks. It is a challenging problem due to the inherent variability of language and the limitations of surface form similarity. Recent pre-trained language models such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) have led to noticeable improvements in semantic similarity detection and subsequent work has explored how these architectures can be further improved. One line of work aims at model compression, making BERT smaller and accessible while mostly preserving its performance (Xu et al., 2020; Goyal et al., 2020; Sanh et al., 2019; Aguilar et al., 2020; Lan et al., 2020; Chen et al., 2020). Other studies seek to further improve model performance by enhancing BERT with external information from knowledge bases (Peters et al., 2019; Wang et al., 2020) or additional modalities (Lu et al., 2019; Lin et al., 2020).

Before the rise of contextualised models, transfer of pre-trained information between datasets and tasks in NLP was based on word embeddings. Over many years, substantial effort was placed into the creation of such embeddings. While originally capturing mainly collocation patterns (Mikolov et al., 2013; Pennington et al., 2014), subsequent work enriched these embeddings with additional information, such as dependencies (Levy and Goldberg, 2014), subword information (Luong et al., 2013; Bojanowski et al., 2017) and semantic lexicons (Faruqui et al., 2015). As a result, there exists a wealth of pre-trained embedding resources for many languages in a unified format which could provide complementary information for contemporary pre-trained contextual models. Moreover, aligning contextual embeddings with static embeddings has shown to increase the performance of the former (Liu et al., 2020).

We propose a new method for injecting pre-trained linguistically-enriched embeddings into any layer of BERT. The model maps any word embeddings into the same space as BERT's hidden representations, then combines them using learned gating parameters. Evaluation of this method on five semantic similarity tasks shows that injecting pre-trained dependency-based and counter-fitted embeddings can further enhance BERT's performance. More specifically, we make the following contributions:

1. We propose GiBERT - a lightweight gated method for injecting externally pre-trained embeddings into BERT (section 3.1).[1]

2. We provide an ablation study and a detailed analysis of the components in the injection architecture (section 5).

3. We demonstrate that the proposed model improves BERT's performance on multiple se-

[1]Code available at https://github.com/wuningxi/GiBERT.

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 2322–2336
November 7–11, 2021.

mantic similarity detection datasets. In comparison to multi-head attention injection, our gated injection method uses fewer parameters while achieving comparable performance for dependency embeddings and improved results for counter-fitted embeddings (section 5).

4. Our qualitative analysis provides insights into GiBERT's improved performance, such as in cases of sentence pairs involving synonyms. (section 5).

2 Related work

BERT modifications Due to BERT's widespread success in NLP, recent studies have focused on further improving BERT by introducing external information. Such work covers a variety of application areas and technical approaches.We broadly categorise such approaches into input-related, external and internal. *Input modifications* (Zhao et al., 2020; Singh et al., 2020; Lai et al., 2020; Ruan et al., 2020) adapt the information that is fed to BERT – e.g. feeding text triples separated by [SEP] tokens instead of sentence pairs as in Lai et al. (2020) – while leaving the architecture unchanged. *Output modifications* (Xuan et al., 2020; Zhang et al., 2020) build on BERT's pre-trained representation by adding external information after the encoding step – e.g. combining it with additional semantic information as in Zhang et al. (2020) – without changing BERT itself. By contrast, *internal modifications* introduce new information directly into BERT by adapting its internal architecture. Fewer studies have taken this approach as this is technically more difficult and might increase the risk of so-called catastrophic forgetting – completely forgetting previous knowledge when learning new tasks (French, 1999; Wen et al., 2018). However, such modifications also offer the opportunity to directly harness BERT's powerful architecture to process the external information alongside the pretrained one. Most existing work on internal modifications has attempted to combine BERT's internal representation with visual and knowledge base information: Lu et al. (2019) modified BERT's transformer block with co-attention to integrate visual and textual information, while Lin et al. (2020) introduced a multimodal model which uses multi-head attention to integrate encoded image and text information between each transformer block. Peters et al. (2019) suggested a word-to-entity attention mechanism to incorporate external knowledge into BERT and Wang et al. (2020) proposed to inject factual and linguistic knowledge through separate adapter modules. Our method introduces external information with an addition-based mechanism which uses fewer parameters than existing attention-based techniques (Lu et al., 2019; Lin et al., 2020; Peters et al., 2019). We further incorporate a gating mechanism to scale injected information so as to reduce the risk of catastrophic forgetting. Moreover, our work investigates the injection of pretrained word embeddings, rather than multimodal or knowledge base information as in previous studies.

Semantic similarity detection Semantic similarity detection is a framework for binary text pair classification tasks such as paraphrase detection, duplicate question identification and answer sentence selection which require detecting the semantic similarity between text pairs (Peinelt et al., 2020). Early semantic similarity methods used feature-engineering techniques, exploring various syntactic (Filice et al., 2017), semantic (Balchev et al., 2016) and lexical features (Tran et al., 2015; Almarwani and Diab, 2017). Subsequent work tried to model text pair relationships either based on increasingly complex neural architectures (Deriu and Cieliebak, 2017; Wang et al., 2017; Tan et al., 2018) or by combining both approaches through hybrid techniques (Wu et al., 2017a; Feng et al., 2017; Koreeda et al., 2017). Most recently, contextual models such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) have reached state-of-the-art performance through pretraining large context-aware language models on vast amounts of textual data. Our study joins up earlier lines of work with current state-of-the-art contextual representations by investigating the combination of BERT with dependency-based and counter-fitted embeddings.

3 GiBERT

3.1 Architecture

We propose GiBERT - a **G**ated **I**njection Method for **BERT**. Our model (Figure 1) is designed with semantic similarity detection in mind and comprises the following: obtaining BERT's intermediate representation from Transformer block i (step 1-2 in Figure 1), creating an alternative input representation based on linguistically-enriched word

embeddings (step 3-4), combining both representations (steps 5-7) and passing on the injected information to subsequent BERT layers to make a final prediction (steps 8-9).

BERT representation We encode a sentence pair with a pre-trained BERT model (Devlin et al. 2019) and obtain BERT's internal representation at different layers (see section 5 for injection layer choices).[2] Following standard practices, we process the two input sentences S_1 and S_2 with a word piece tokenizer (Wu et al., 2017b) and combine them using '[CLS]' and '[SEP]' tokens, which indicate sentence boundaries. The word pieces are then mapped to ids, resulting in a sequence of word piece ids $\mathbf{E^W} = [w_1, ..., w_N]$ where N indicates the number of word pieces in the sequence (step 1 in Figure 1). In the case of embedding layer injection, we use BERT's embedding layer output denoted with $\mathbf{H^0}$ which results from summing the word piece embeddings $\mathbf{E^W}$, positional embeddings $\mathbf{E^P}$ and segment embeddings $\mathbf{E^S}$ (step 2):

$$\begin{aligned} \mathbf{H^0} &= \text{LayerNorm}(\mathbf{E^W} + \mathbf{E^P} + \mathbf{E^S}) \\ \mathbf{E^W}, \mathbf{E^P}, \mathbf{E^S}, \mathbf{H^0} &\in \mathbb{R}^{N \times D} \end{aligned} \quad (1)$$

where D is the internal hidden size of BERT ($D = 768$ for $\text{BERT}_{\text{BASE}}$). For injecting information at later layers, we obtain BERT's internal representation $\mathbf{H^i} \in \mathbb{R}^{N \times D}$ after transformer block i with $1 \leq i \leq L$ (step 2):

$$\begin{aligned} \mathbf{M^i} &= \text{LayerNorm}(\mathbf{H^{i-1}} + \text{MultiheadAtt}(\mathbf{H^{i-1}})) \\ \mathbf{H^i} &= \text{LayerNorm}(\mathbf{M^i} + \text{FeedForward}(\mathbf{M^i})) \end{aligned} \quad (2)$$

where L is the number of Transformer blocks ($L = 12$ for $\text{BERT}_{\text{BASE}}$) and MultiheadAtt denotes multihead attention.

External embedding representation To enrich this representation, we obtain alternative representations for the tokens in S_1 and S_2 by looking up word embeddings in a pre-trained embedding matrix $\mathbf{E} \in \mathbb{R}^{|V| \times E}$, where $|V|$ denotes vocabulary size and E the dimensionality of the pre-trained embeddings (step 3, section 3.2 presents details regarding our choice of pre-trained embeddings). In order to map word embedding representations to BERT's word piece representations, an alignment function duplicates the word embedding for the corresponding number of subwords, then adds BERT's special '[CLS]' and '[SEP]' tokens, resulting in an injection sequence $\mathbf{I} \in \mathbb{R}^{P \times E}$ (step 4). For example, it assigns the pre-trained embedding of the word 'prompt' to both of the corresponding word pieces 'pro' and '##mpt' (see Figure 1).

Attention injection Multihead attention was proposed by Vaswani et al. (2017):

$$\begin{aligned} &\text{MultiheadAtt}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = [\text{head}_1; ...; \text{head}_h]\mathbf{W^O} \\ &\text{head}_j = \text{Attention}(\mathbf{QW_j^Q}, \mathbf{KW_j^K}, \mathbf{VW_j^V}) \end{aligned} \quad (3)$$

and is employed in Transformer networks in the form of self-attention (where queries $\mathbf{Q}$, keys $\mathbf{K}$ and values $\mathbf{V}$ come from the previous layer) or encoder-decoder attention (where queries come from the decoder, keys and values from the encoder). Previous work has successfully employed multihead attention to combine BERT with external information (see section 2). In their multimodal VilBERT model, Lu et al. (2019) combined textual and visual representations by passing the keys and values from each modality as input to the other modality's multi-head attention block. Similarly, Peters et al. (2019) used multihead attention to combine projected BERT representations (as queries) with entity-span representations (as keys and values) in their knowledge-enrichment method for BERT. For our case of combining BERT with the injection sequence, we can therefore experiment with the following multi-head attention injection method:

$$\mathbf{H^{i'}} = \mathbf{H^i} + \text{MultiHeadAtt}(\mathbf{H^i}, \mathbf{I}, \mathbf{I}) \quad (4)$$

where queries are provided by BERT's internal representation, while keys and values come from the injected embeddings. The output of the attention mechanism is then combined with the previous layer through addition.

Gated injection We also propose an alternative method for combining external embeddings with BERT which requires only 14% of parameters used in multi-head attention (0.23M instead of 1.64M, see Appendix G). First, we add a feed-forward layer – consisting of a linear layer with weights $\mathbf{W^P} \in \mathbb{R}^{D \times E}$ and bias $\mathbf{b^P} \in \mathbb{R}^D$ with a tanh activation function – to project the aligned embedding sequence to BERT's internal dimensions and squash the output values to a range between -1 and 1 (step 5):

$$\mathbf{P} = \text{FeedForward}(\mathbf{I}) \in \mathbf{R}^{N \times D} \quad (5)$$

[2] We use the uncased version of $\text{BERT}_{\text{BASE}}$ available through Tensorflow Hub.

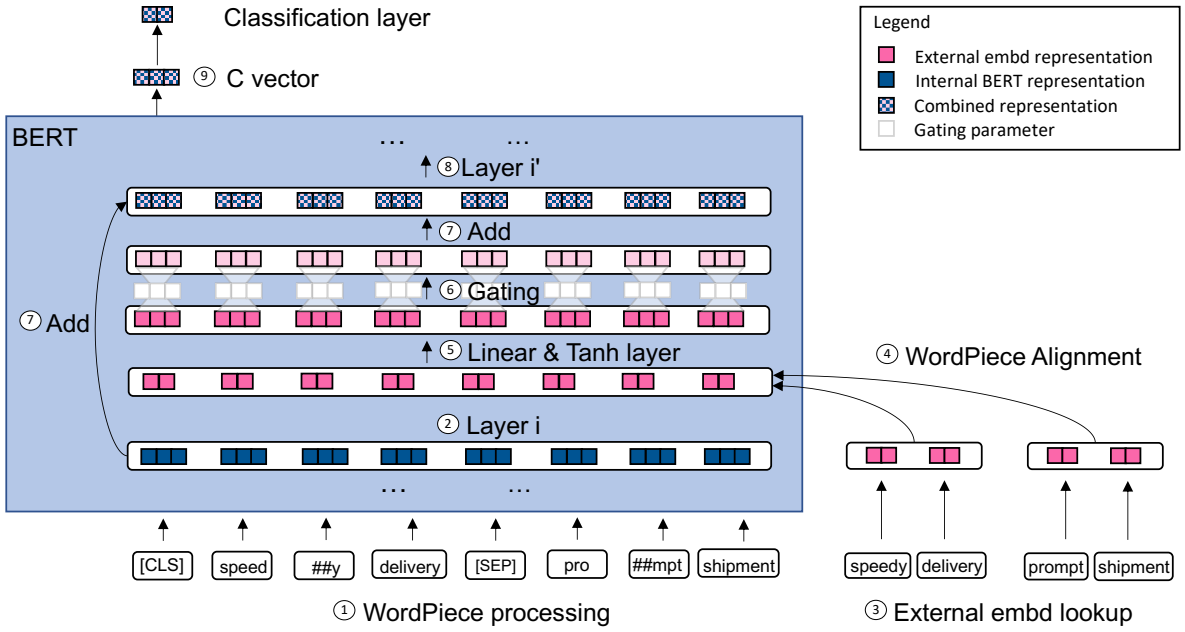

Figure 1: Our proposed GiBERT architecture illustrated with a toy example (where internal BERT dimension $d = 3$ and embedding dimension $e = 2$). The input consists of a sentence pair which is processed with a word piece tokenizer (step 1) and encoded with BERT up to layer i (step 2). We obtain an alternative representation for the sentences based on pretrained word embeddings (step 3), while ensuring that external word embeddings are aligned with BERT's word pieces by repeating embeddings for tokens which have been split into several word pieces (step 4). The aligned word embedding sequence is passed through a linear and tanh layer to match BERT's embedding dimension (step 5). We apply a gating mechanism (step 6) before adding the injected information to BERT's representation from layer i (step 7). The combined representation is passed to the next layer (step 8). At the final layer, the C vector is used as the sentence pair representation, followed by a classification layer (step 9).

Then, we use a residual connection to inject the projected external information into BERT's representation from Transformer block i (see section 5 for injection at different locations) and obtain a new enriched representation $\mathbf{H^{i'}} \in \mathbb{R}^{N \times D}$:

$$\mathbf{H^{i'}} = \mathbf{H^{i}} + \mathbf{P} \tag{6}$$

However, injection values in $\mathbf{P}$ can range between -1 and 1, whereas values in BERT's internal representation $\mathbf{H^{i}}$ usually range from -0.1 to 0.1. When external information is directly injected using an additive operation, BERT's pre-trained information can be easily overwritten by the injection, resulting in catastrophic forgetting. To address this potential pitfall, we further propose a gating mechanism which uses a gating vector $\mathbf{g} \in \mathbb{R}^{D}$ to scale the injected information before combining it with BERT's internal representation as follows:

$$\mathbf{H^{i'}} = \mathbf{H^{i}} + \mathbf{g} \odot \mathbf{P} \tag{7}$$

where $\odot$ denotes element-wise multiplication using broadcasting (step 6 & 7). The gating parameters are initialised with zeros and updated during training. This has the benefit of starting fine-tuning from representations which are equivalent to vanilla BERT and gradually introducing the injected information during fine-tuning along certain dimensions. If specific features in the external representations are not beneficial for the task, it is easy for the model to ignore them by keeping the gating parameters at zero.

Output layer The combined representation $\mathbf{H^{i'}}$ is then fed to BERT's next Transformer block $i + 1$ (step 8). At the final Transformer block $\mathbf{L}$, we use the $\mathbf{c} \in \mathbb{R}^{D}$ vector which corresponds to the '[CLS]' token in the input and is typically used as the sentence pair representation (step 9). As proposed by Devlin et al. (2019), this is followed by a softmax classification layer (with weights $\mathbf{W^L} \in \mathbb{R}^{C \times D}$ and $\mathbf{b^L} \in \mathbb{R}^{C}$) to calculate class probablilities where C indicates the number of classes. During finetuning, we train the entire model for 3 epochs with early stopping and cross-entropy loss. Learning rates are tuned for each seed and dataset based on development set performance (reported in Appendix D).

3.2 Injected Embeddings

While many different embedding resources exist, here we focus on experimenting with pre-trained word representations that are beneficial for the semantic similarity detection tasks and also contain information complementary to BERT. Embeddings such as word2vec and Glove leverage co-occurrence patterns which have been shown to be also captured by BERT (Gan et al., 2020). Recent contextualised embeddings risk redundancy with BERT due to the similarity of used approaches. We reason that linguistically-enriched embeddings are most likely to be complementary to BERT, as the model has not been explicitly trained on semantic or syntactic resources and has only partial knowledge of syntax and semantics (Rogers et al., 2020). We hence experiment with injecting dependency-based (Levy and Goldberg, 2014) and counter-fitted embeddings (Mrkšić et al., 2016) into BERT, which have been found useful for semantic similarity modelling and other related tasks (Filice et al., 2017; Feng et al., 2017; Alzantot et al., 2018; Jin et al., 2020).

The 300-dim dependency-based embeddings by Levy and Goldberg (2014) extend the SkipGram embedding algorithm proposed by Mikolov et al. (2013) by replacing linear bag-of-word contexts with dependency-based contexts which are extracted from parsed English Wikipedia sentences. As BERT has not been exposed to dependencies during pretraining and previous studies have found that BERT's knowledge of syntax is only partial (Rogers et al., 2020), we reason that these embeddings could provide complementary information.

The 300-dim counter-fitted embeddings by Mrkšić et al. (2016) integrate antonymy and synonymy relations into word embeddings based on an objective function which combines three principles: repelling antonyms, attracting synonyms and preserving the vector space. For training, they obtain synonym and antonym pairs from the Paraphrase Database and WordNet, demonstrating an increased performance on SimLex-999 (Hill et al., 2015). We use their highest-scoring vectors which were obtained by applying the counter-fitting method to Paragram vectors from Wieting et al. (2015). Antonym and synonym relations are particularly important for paraphrase detection and injecting them into BERT gives the model access to this useful additional information.

4 Evaluation

4.1 Datasets and Tasks

We focus on semantic similarity detection which is a fundamental problem in NLP and involves modelling the semantic relationship between two sentences in a binary classification setup. We work with the following five widely used English language datasets which cover a range of sizes and tasks (including paraphrase detection, duplicate question identification and answer sentence selection, see Appendix A for details).

MSRP The Microsoft Research Paraphrase dataset (MSRP) contains 5K pairs of sentences from news websites which were collected based on heuristics and an SVM classifier. Gold labels are based on human binary annotations for sentential paraphrase detection (Dolan and Brockett, 2005).

SemEval The SemEval 2017 CQA dataset (Nakov et al., 2017) consists of three subtasks involving posts from the online forum Qatar Living[3]. Each subtask provides an initial post as well as 10 posts which were retrieved by a search engine and annotated with binary labels by humans. The task requires the distinction between relevant and non-relevant posts. The original problem is a ranking setting, but since the gold labels are binary, we focus on a classification setup. In **subtask A**, the posts are questions and comments from the same thread, in an answer sentence selection setup (26K instances). **Subtask B** is question paraphrase detection (4K instances). **Subtask C** is similar to A but comments were retrieved from an external thread (47K). We use the 2016 test set as the dev set and the 2017 test set as the test set.

Quora The Quora duplicate questions dataset is the largest of the selected datasets, consisting of more than 400K question pairs with binary labels.[4] The task is to predict whether two questions are duplicates. We use Wang et al. (2017)'s train/dev/test set partition.

All of the above datasets provide two short texts, each usually a single sentence but sometimes consisting of multiple sentences. For simplicity, we refer to each short text as 'sentence'. We frame the

[3] https://www.qatarliving.com/

[4] https://engineering.quora.com/Semantic-Question-Matching-with-Deep-Learning

task as semantic similarity detection between two sentences through binary classification.

4.2 Metrics

Our main evaluation metric is the F1 score as this is more meaningful than accuracy for datasets with imbalanced label distributions (such as SemEval C, see Appendix A). We also report performance on difficult cases using the non-obvious F1 score (Peinelt et al., 2019). This metric distinguishes obvious from non-obvious instances in a dataset based on lexical overlap and gold labels, and calculates a separate F1 score for challenging cases. This value therefore tends to be lower than the regular F1 score. Dodge et al. (2020) recently showed that early stopping and random seeds can have considerable impact on the performance of finetuned BERT models, therefore we finetune all models for 3 epochs with early stopping (based on dev F1) and report average model performance across two different seeds. Hyperparameter settings of all BERT-based models are identical, except for learning rate and injection location which are tuned with grid search, see Appendix D.

4.3 Baselines

SemEval systems We compare against the best SemEval 2017 system for each subtask based on F1 score: KeLP (Filice et al., 2017), ECNU (Wu et al., 2017a) and Bunji (Koreeda et al., 2017).

BERT Following standard practice, we encode the sentence pair with BERT's C vector from the final layer, followed by a softmax layer as proposed by Devlin et al. (2019). We use Tensorflow Hub's distribution of $\text{BERT}_{\text{BASE}}$.

SemBERT Additionally we compare with the semantics-aware BERT model (SemBERT, Zhang et al. 2020) which uses a semantic role labeler. As the original paper reports results on different dataset versions, we ran the official code on our datasets. The longer sentences in SemEval could not fit on a single GPU due to the larger model size.

tBERT We also combine embeddings with BERT using an averaging and concatenation method proposed in tBERT (Peinelt et al., 2020). Instead of the word topics in the original system, we use pretrained counter-fitted and dependency embeddings for direct comparison with our methods.

AiBERT We further provide an alternative **A**ttention-based embedding **I**njection method for **BERT** based on the multihead attention injection mechanism described in equations 3 to 4. Following the same procedure as GiBERT, we tune the injection location (see Appendix E).

5 Results

Full model GiBERT with counter-fitted embeddings outperforms all other systems in both average F1 and average non-obvious F1 score (see Table 1). This shows that the model improves on challenging dataset instances, rather than merely leveraging shallow surface patterns. It is worth noting that

	F1						**non-obvious F1**					
	MSRP	Quora	SemEval A	SemEval B	SemEval C	avg	MSRP	Quora	SemEval A	SemEval B	SemEval C	avg
Previous systems												
KeLP◇	-	-	-	.506	-	-	-	-	-	.199	-	-
ECNU◇	-	-	.777	-	-	-	-	-	.707	-	-	-
Bunji◇	-	-	-	-	.197	-	-	-	-	-	.028	-
BERT⋆	.876	.902	.704	.473	.268	.645	.827	.860	.656	.243	.085	.534
SemBERT⋆	.876	.901	✗	✗	✗	-	.820	**.874**	✗	✗	✗	-
$\text{tBERT}_{\text{dependency}}$⋆	.882	.906	**.780**	.510	.242	.664	.827	.858	**.728**	.262	**.090**	.553
$\text{tBERT}_{\text{counter-fitted}}$⋆	.879	.906	.756	.500	.215	.651	.824	.857	.699	.258	.064	.540
Our implementation												
$\text{AiBERT}_{\text{dependency}}$	.863	.903	.738	.498	**.282**	.657	.792	.866	.681	**.268**	**.090**	.539
$\text{AiBERT}_{\text{counter-fitted}}$	.877	.904	.724	.496	.263	.653	.835	.867	.662	.264	.076	.541
$\text{GiBERT}_{\text{dependency}}$	.883	.904	.768	.474	.238	.653	.849	.864	.704	.231	.087	.547
$\text{GiBERT}_{\text{counter-fitted}}$	**.884**	**.907**	**.780**	**.511**	.256	**.668**	**.858**	.862	.719	.248	**.090**	**.555**

Table 1: Model performance on test set. All BERT-based methods use $\text{BERT}_{\text{BASE}}$. Bold font highlights the best result overall, our best systems are underlined. avg = average performance across all datasets, ◇ = results from publication, ⋆ = official code run on our data, ✗= run failed due to insufficient GPU memory, $_{\text{dep}}$ = dependency embeddings, $_{\text{counter}}$ = counter-fitted embeddings.

GiBERT has the fewest parameters of all BERT-enhancing models (see Appendix F) and doesn't require any additional preprocessing tools (such as the neural SRL tagger required by SemBERT), making it more efficient than SemBERT, tBERT and AiBERT. The largest improvement of GiBERT over BERT is observed with counter-fitted embeddings, especially on SemEval A and B (the datasets with the highest proportion of examples involving synonym pairs, see Table 5). GiBERT with dependency embeddings still improves over vanilla BERT, but gains tend to be smaller and roughly similar to the more complex AiBERT injection method. tBERT always combines external information with BERT at the latest possible stage, which is beneficial for dependency embeddings but less effective for counter-fitted embeddings (compare Table 3). Compared to GiBERT, this makes it a less flexible method while also requiring more parameters. Our results indicate that semantic information is more important for the tasks at hand and syntactic information benefits from a late integration.

Gating mechanism Catastrophic forgetting is a potential problem when introducing external information into a pre-trained model as the injected information could disturb or completely overwrite existing knowledge (Wang et al., 2020). In our proposed model, a gating mechanism is used to scale injected embeddings before adding them to the pre-trained internal BERT representation (see section 3.1). To understand the importance of this mechanism, in Table 2 we contrast development set performance for injecting information after the embedding layer with gating – as defined in equation 7 – and without – as in equation 6. For dependency embedding injection without gating, performance only improves on 2 out of 5 datasets over the baseline and in some cases even drops below BERT's performance, while it outperforms the baseline on all datasets when using the gating mechanism.

	MSRP	Quora	SemEval A	SemEval B	SemEval C
BERT	.906	.906	.714	.754	.414
GiBERT with dependency embeddings					
- no gating	.906	.905	.732	.751	.424
- with gating	**.913**	**.908**	**.755**	**.778**	**.433**
GiBERT with counter-fitted embeddings					
- no gating	**.907**	.906	.733	.763	.435
- with gating	**.907**	**.908**	**.751**	**.767**	**.451**

Table 2: Development set F1 scores of GiBERT models injecting pretrained embeddings after the embedding layer with vs. without gating mechanism.

	MSRP	Quora	SemEval A	SemEval B	SemEval C
BERT	.906	.906	.714	.754	.414
GiBERT with dependency embeddings					
- embd layer	.913	.908	.755	**.778**	.433
- layer 6	.911	.908	.755	.776	.438
- layer 11	**.914**	**.910**	**.760**	.773	**.444**
GiBERT with counter-fitted embeddings					
- embd layer	.907	.908	.751	.767	.451
- layer 6	**.917**	**.909**	**.760**	**.771**	**.464**
- layer 11	.910	.907	.755	**.771**	.450

Table 3: Development set F1 scores of embedding injection at different layers.

Counter-fitted embedding injection without gating improves on 4 out of 5 datasets, with further improvements when adding gating, outperforming the vanilla BERT model across all datasets. In addition, gating makes model training more stable and reduces failed runs (where the model predicted only the majority class) on the particularly imbalanced SemEval C dataset. This highlights the importance of the gating mechanism in our proposed method.

Injection location In our proposed model, information can be injected between any of BERT's pre-trained transformer blocks. We reason that different locations may be more appropriate for certain kinds of embeddings as previous research has found that different types of information tend to be encoded and processed at specific BERT layers (Rogers et al., 2020). We experiment with three locations: after the embedding layer (using $\mathbf{H^0}$), after the middle layer (using $\mathbf{H^6}$ in $\text{BERT}_{\text{BASE}}$) and after the penultimate layer (using $\mathbf{H^{11}}$ in $\text{BERT}_{\text{BASE}}$). Table 3 shows that midlayer injection is ideal for counter-fitted embeddings, while late injection appears to work best for dependency embeddings (Table 3). This is in line with previous work which found that BERT tends to processes syntactic information at later layers than linear word-level information (Rogers et al., 2020). We consequently use these injection locations in our final model (see Appendix E for AiBERT's tuned injection locations).

Error Analysis Counter-fitted embeddings are designed to explicitly encode synonym and antonym relationships between words. To better understand how the injection of counter-fitted embeddings affects the ability of our model to deal

	Sentence 1	Sentence 2	Gold label	BERT prediction	GiBERT prediction
(1)	it took me more than 10 people; over the course of the whole day to convince my point at qatar **airways**... as to how my points needs to be redeemed... at long last my point was made... dont seem know what they are doing??? appalling to say the least	this isn't the first time. so many rants by irate customers on so many diverse situations signals a very serious problem. so called first class **airlines** and no basic customer care. over confidence much?	is related	not related	is related
(2)	hi; my **wife** was on a visit visa; today; her residency visa was issued; so i went to immigration and paid 500 so there is no need to leave the country and enter again on the residency visa . she has done her medical before for the visit visa extension; do we need to do the medical again for the residency visa? thanks	dear all; please let me know how many days taking for approve family visa nw; am last wednesday (12/09/2012) apply family visa for my **husband** and daughter; but still now showing in moi website itz under review; itz usual reply? why delayed like this? please help me regards divya	is related	is related	not related

Table 4: Examples from the Semeval development set. Synonym and antonym pairs are highlighted in bold.

with instances involving such semantic relations, we use synonym and antonym pairs from the PPDB and Wordnet (provided by Mrkšić et al. 2016) and search the development partition of the datasets for sentence pairs where the first sentence contains one word of the synonym/antonym pair and the second sentence the other word. Table 5 reports F1 performance of our model on cases with synonym pairs, antonym pairs and neither one. We find that our model's F1 performance particularly improves over BERT on instances containing synonym pairs, as illustrated in example (1) in Table 4. By contrast, the performance on cases with antonym pairs stays roughly the same, although slightly decreasing on Quora. As illustrated by example (2) in Table 4, word pairs can be antonyms in isolation (e.g. husband - wife), but not in the specific context of a given example. In rare cases, the injection of distant antonym pair embeddings can therefore deter the model from detecting related sentence pairs. We also observe a slight performance boost for cases without synonym or antonym pairs which could be due to improved representations for words which occurred in examples without their synonym or antonym counterpart.

	MSRP	Quora	SemEval		
			A	B	C
	Instances with **antonym** pairs				
	(4%)	(4%)	(21%)	(28%)	(20%)
BERT	**.81**	**.87**	**.77**	**.75**	**.46**
GiBERT	**.81**	.86	**.77**	**.75**	**.46**
	Instances with **synonym** pairs				
	(11%)	(9%)	(22%)	(31%)	(17%)
BERT	.87	.90	.81	.78	**.54**
GiBERT	**.90**	**.91**	**.82**	**.83**	**.54**
	Instances **without** synonym/antonym pairs				
	(85%)	(87%)	(64%)	(51%)	(68%)
BERT	.91	**.91**	.71	.72	.36
GiBERT	**.92**	**.91**	**.73**	**.73**	**.41**

Table 5: F1 score on instances containing synonymy pairs, antonymy pairs or no pairs across datasets.

6 Conclusion

In this paper, we introduced a new approach for injecting external information into BERT. Our proposed method adds linguistically enriched embeddings to BERT's internal representation through a lightweight gating mechanism which requires fewer parameters than previous approaches. Evaluating our injection method on multiple semantic similarity detection datasets, we demonstrated that injecting counter-fitted embeddings clearly improved performance over vanilla BERT and on average outperformed all baselines on the task, while dependency embedding injection achieved slightly smaller gains. In comparison to the multi-head attention injection mechanism, we found the gated method at least as effective, with comparable performance for dependency embeddings and improved results for counter-fitted embeddings. In ablation studies, we showed that the choice of injection location and the use of the proposed gating mechanism are crucial for our architecture. Our qualitative analysis highlighted that counter-fitted injection was particularly helpful for instances involving synonym pairs. Future work could explore combining multiple embedding sources or injecting other types of information. Another direction is to investigate the usefulness of embedding injection for other tasks.

Acknowledgments

This work was supported by Microsoft Azure and The Alan Turing Institute under the EPSRC grant EP/N510129/1. It was also supported by a UKRI/EPSRC Turing AI Fellowship to Maria Liakata (grant no. EP/V030302/1).

References

Gustavo Aguilar, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. 2020. Knowledge Distillation from Internal Representations. *arXiv:1910.03723 [cs]*.

Nada Almarwani and Mona Diab. 2017. GW_QA at SemEval-2017 Task 3- Question Answer Reranking on Arabic Fora. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017*, pages 344–348, Vancouver, Canada. Association for Computational Linguistics.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating Natural Language Adversarial Examples. *arXiv:1804.07998 [cs]*.

Daniel Balchev, Yasen Kiprov, Ivan Koychev, and Preslav Nakov. 2016. PMI-cool at SemEval-2016 Task 3: Experiments with PMI and Goodness Polarity Lexicons for Community Question Answering. In *SemEval@ NAACL-HLT*, pages 844–850.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics (TACL)*, 5:135–146.

Daoyuan Chen, Yaliang Li, Minghui Qiu, Zhen Wang, Bofang Li, Bolin Ding, Hongbo Deng, Jun Huang, Wei Lin, and Jingren Zhou. 2020. AdaBERT: Task-Adaptive BERT Compression with Differentiable Neural Architecture Search. *arXiv:2001.04246 [cs]*.

Jan Milan Deriu and Mark Cieliebak. 2017. SwissAlps at SemEval-2017 Task 3: Attention-based Convolutional Neural Network for Community Question Answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation.*, volume 17, pages 334–338, Vancouver, Canada. 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. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)*, pages 4171—4186, Minneapolis, USA. Association for Computational Linguistics.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-Tuning Pretrained Language Models: Weight Initializations, Data Orders, and Early Stopping. *arXiv:2002.06305 [cs]*.

William B. Dolan and Chris Brockett. 2005. Automatically Constructing a Corpus of Sentential Paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP@IJCNLP)*, pages 9–16, Jeju Island, Korea. Asian Federation of Natural Language Processing.

Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting Word Vectors to Semantic Lexicons. In *The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2015)*, pages 1606–1615, Denver, USA. Association for Computational Linguistics.

Wenzheng Feng, Yu Wu, Wei Wu, Zhoujun Li, and Ming Zhou. 2017. Beihang-MSRA at SemEval-2017 Task 3- A Ranking System with Neural Matching Features for Community Question Answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017*, pages 280–286, Vancouver, Canada. Association for Computational Linguistics.

Simone Filice, Giovanni Da San Martino, and Alessandro Moschitti. 2017. KeLP at SemEval-2017 Task 3- Learning Pairwise Patterns in Community Question Answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017*, pages 326–333, Vancouver, Canada. Association for Computational Linguistics.

Robert French. 1999. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135.

Leilei Gan, Zhiyang Teng, Yue Zhang, Linchao Zhu, Fei Wu, and Yi Yang. 2020. SemGloVe: Semantic Co-occurrences for GloVe from BERT. *arXiv:2012.15197 [cs]*.

Saurabh Goyal, Anamitra Roy Choudhary, Venkatesan Chakaravarthy, Saurabh ManishRaje, Yogish Sabharwal, and Ashish Verma. 2020. PoWER-BERT: Accelerating BERT inference for Classification Tasks. *arXiv:2001.08950 [cs, stat]*.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating Semantic Models With (Genuine) Similarity Estimation. *Computational Linguistics*, 41(4):665–695.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. *arXiv:1907.11932 [cs]*.

Yuta Koreeda, Takuya Hashito, Yoshiki Niwa, Misa Sato, Toshihiko Yanase, Kenzo Kurotsuchi, and Kohsuke Yanai. 2017. Bunji at SemEval-2017 Task 3: Combination of Neural Similarity Features and Comment Plausibility Features. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval@ACL 2017)*, pages 353–359, Vancouver, Canada. Association for Computational Linguistics.

Tuan Manh Lai, Quan Hung Tran, Trung Bui, and Daisuke Kihara. 2020. A Simple but Effective BERT Model for Dialog State Tracking on Resource-Limited Systems. *arXiv:1910.12995 [cs]*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *arXiv:1909.11942 [cs]*.

Omer Levy and Yoav Goldberg. 2014. Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308, Baltimore, USA. Association for Computational Linguistics.

Junyang Lin, An Yang, Yichang Zhang, Jie Liu, Jingren Zhou, and Hongxia Yang. 2020. InterBERT: Vision-and-Language Interaction for Multi-modal Pretraining. *arXiv:2003.13198 [cs]*.

Qianchu Liu, Diana McCarthy, and Anna Korhonen. 2020. Towards Better Context-aware Lexical Semantics: Adjusting Contextualized Representations through Static Anchors. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 4066–4075.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. *arXiv:1908.02265 [cs]*.

Thang Luong, Richard Socher, and Christopher Manning. 2013. Better Word Representations with Recursive Neural Networks for Morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning, CoNLL 2013*, pages 104–113, Sofia, Bulgaria. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations (ICLR 2013)*, Scottsdale, USA.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting Word Vectors to Linguistic Constraints. *arXiv:1603.00892 [cs]*.

Preslav Nakov, Doris Hoogeveen, Llúis Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 Task 3: Community Question Answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval@ACL 2017)*, pages 27–48, Vancouver, Canada. Association for Computational Linguistics.

Nicole Peinelt, Maria Liakata, and Dong Nguyen. 2019. Aiming beyond the Obvious: Identifying Non-Obvious Cases in Semantic Similarity Datasets. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2792–2798, Florence, Italy. Association for Computational Linguistics.

Nicole Peinelt, Dong Nguyen, and Maria Liakata. 2020. tBERT: Topic Models and BERT Joining Forces for Semantic Similarity Detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7047–7055, Seattle, USA. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 2227–2237, New Orleans, USA. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Robert L. Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge Enhanced Contextual Word Representations. *arXiv:1909.04164 [cs]*.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A Primer in BERTology: What we know about how BERT works. *arXiv:2002.12327 [cs]*.

Yu-Ping Ruan, Zhen-Hua Ling, Jia-Chen Gu, and Quan Liu. 2020. Fine-Tuning BERT for Schema-Guided Zero-Shot Dialogue State Tracking. *arXiv:2002.00181 [cs]*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv:1910.01108 [cs]*.

Gaurav Singh, Zahra Sabet, John Shawe-Taylor, and James Thomas. 2020. Constructing Artificial Data for Fine-tuning for Low-Resource Biomedical Text

Tagging with Applications in PICO Annotation. *arXiv:1910.09255 [cs, stat]*.

Chuanqi Tan, Furu Wei, Wenhui Wang, Weifeng Lv, and Ming Zhou. 2018. Multiway Attention Networks for Modeling Sentence Pairs. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4411–4417, Stockholm, Sweden.

Quan Hung Tran, Vu Tran, Tu Vu, Minh Nguyen, and Son Bao Pham. 2015. JAIST: Combining Multiple Features for Answer Selection in Community Question Answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 215–219, Denver, USA. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, pages 5998–6008, Long Beach, USA.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu ji, Cuihong Cao, Daxin Jiang, and Ming Zhou. 2020. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. *arXiv:2002.01808 [cs]*.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral Multi-Perspective Matching for Natural Language Sentences. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4144–4150, Melbourne, Australia.

Junfeng Wen, Yanshuai Cao, and Ruitong Huang. 2018. Few-Shot Self Reminder to Overcome Catastrophic Forgetting. *arXiv:1812.00543 [cs, stat]*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. From Paraphrase Database to Compositional Paraphrase Model and Back. *Transactions of the Association for Computational Linguistics*, 3:345–358.

Guoshun Wu, Yixuan Sheng, Man Lan, and Yuanbin Wu. 2017a. ECNU at SemEval-2017 Task 3- Using Traditional and Deep Learning Methods to Address Community Question Answering Task. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval@ACL 2017)*, pages 356–360, Vancouver, Canada. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017b. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *Transactions of the Association for Computational Linguistics*, pages 339–351.

Canwen Xu, Wangchunshu Zhou, Tao Ge, Furu Wei, and Ming Zhou. 2020. BERT-of-Theseus: Compressing BERT by Progressive Module Replacing. *arXiv:2002.02925 [cs]*.

Zhenyu Xuan, Chuyu Ma, and Shengyi Jiang. 2020. FGN: Fusion Glyph Network for Chinese Named Entity Recognition. *arXiv preprint: 2001.05272*.

Zhuosheng Zhang, Yuwei Wu, Hai Zhao, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiang Zhou. 2020. Semantics-aware BERT for Language Understanding. *arXiv:1909.02209 [cs]*.

Lingyun Zhao, Lin Li, and Xinhao Zheng. 2020. A BERT based Sentiment Analysis and Key Entity Detection Approach for Online Financial Texts. *arXiv:2001.05326 [cs]*.

Appendix

A Dataset Characteristics

Dataset	Language	Source	Positive instances	Train	Dev	Test	Total size
MSRP	English	News websites	67%	3576	500	1725	5k
Quora	English	Q&A website	37%	384347	9999	9999	404k
SemEval A	English	Online forum	39%	20340	3270	2930	26K
SemEval B	English	Online forum	36%	3169	700	880	4K
SemEval C	English	Online forum	9%	31690	7000	8800	47K

Table 6: Properties of selected semantic similarity detection data sets.

B Examples

Dataset	Task	Example		
		Sentence 1	Sentence 2	Label
Quora	Duplicate question identification	There are only 2,000 Roman Catholics living in Banja Luka now.	There are just a handful of Catholics left in Banja Luka.	is_paraphrase
MSRP	Paraphrase detection	Which is the best way to learn coding?	How do you learn to program?	is_paraphrase
SemEval	(A) Internal answer sentence selection	Anybody recommend a good dentist in Doha?	Dr Sarah Dental Clinic	is_related
	(B) Question paraphrase detection	Where I can buy good oil for massage?	Blackheads - Any suggestions on how 0 to get rid of them??	not_related
	(C) External answer sentence selection	Can anybody tell me where is Doha clinic?	Dr. Rizwi - Al Ahli Hospital	not_related

Table 7: Text pair similarity data sets with examples.

C Preprocessing

We lowercase and tokenise all datasets, replacing images and URLs with placeholders. Sequences exceeding the maximum length are cut off. Each model is trained on a single NVIDIA Tesla K80 GPU.

D Hyper-Parameters

For datasets with long sentences, the batch size is reduced from 32 to 16 to fit on a single GPU. We tune learning rates (2e-5, 3e-5 and 5e-5) through grid search based on highest development set F1 scores. The best injection location (embd layer, layer 6 and layer 11) is selected based on best development set F1 scores across datasets (see Tables 2 and 9). Table 8 shows the final hyperparameters after tuning. In total, we train 6 BERT, SemBERT and tBERT models (2 different random seeds x 3 learning rates), as well as 18 AiBERT and GiBERT models (2 different random seeds x 3 learning rates x 3 injection locations) for each dataset.

	MSRP	Quora	SemEval		
			A	B	C
All models					
Batch size	32	32	16	32	16
Max length	80	48	300	200	300
BERT					
LR (1st seed)	5e-5	2e-5	3e-5	2e-5	2e-5
LR (2nd seed)	5e-5	2e-5	2e-5	2e-5	3e-5
SemBERT					
Max # of PAS	3	3			
LR (1st seed)	2e-5	5e-5			
LR (2nd seed)	3e-5	5e-5			
tBERT with dependency-based embeddings					
Hidden layers	1	1	1	1	1
LR (1st seed)	2e-5	2e-5	2e-5	2e-5	2e-5
LR (2nd seed)	5e-5	2e-5	3e-5	3e-5	2e-5
BERT with counter-fitted embeddings					
Hidden layers	1	1	1	1	1
LR (1st seed)	2e-5	5e-5	2e-5	2e-5	2e-5
LR (2nd seed)	5e-5	2e-5	2e-5	3e-5	2e-5
AiBERT with dependency-based embeddings					
Location	6	6	6	6	6
LR (1st seed)	3e-5	3e-5	2e-5	3e-5	2e-5
LR (2nd seed)	5e-5	2e-5	2e-5	5e-5	2e-5
AiBERT with counter-fitted embeddings					
Location	6	6	6	6	6
LR (1st seed)	5e-5	2e-5	2e-5	3e-5	2e-5
LR (2nd seed)	5e-5	3e-5	5e-5	3e-5	2e-5
GiBERT with dependency-based embeddings					
Location	11	11	11	11	11
LR (1st seed)	2e-5	3e-5	2e-5	3e-5	2e-5
LR (2nd seed)	3e-5	2e-5	2e-5	5e-5	3e-5
GiBERT with counter-fitted embeddings					
Location	6	6	6	6	6
LR (1st seed)	5e-5	2e-5	2e-5	5e-5	2e-5
LR (2nd seed)	5e-5	3e-5	3e-5	5e-5	3e-5

Table 8: Tuned hyper-parameters for BERT-based models. LR = learning rate. Location = injection location, length = sequence length in tokens, PAS = predicate-argument structures.

E Injection Location for AiBERT

Based on the development set results shown in Table 9, the final AiBERT model uses injection layer 6 for both dependency and counter-fitted embeddings.

	MSRP	Quora	SemEval		
			A	B	C
AiBERT with dependency embeddings					
- embd layer	.9028	.9058	.7306	**.7512**	.4214
- layer 6	**.9040**	**.9073**	**.7342**	.7493	.4177
- layer 11	.8993	.9070	.7265	.7506	**.4296**
AiBERT with counter-fitted embeddings					
- embd layer	.9008	**.9078**	.7181	.7472	.4186
- layer 6	**.9066**	.9070	**.7269**	.7500	**.4267**
- layer 11	.9010	.9068	.7171	**.7505**	.4206

Table 9: Development F1 scores of attention-based embedding injection at different layers.

F Total Parameters

	Total parameters
BERT	110.1M
GiBERT	110.3M
tBERT	111.0M
AiBERT	111.7M
SemBERT	111.9M

Table 10: BERT-based models ordered by size (using $\text{BERT}_{\text{BASE}}$).

G Required Injection Parameters

This section compares the number of required parameters in the two alternative injection methods discussed in section 4.1: a multihead attention injection mechanism (AiBERT) and a novel lightweight gated injection mechanism (GiBERT).

Attention injection In multihead attention injection (equations 3 to 4), the keys are provided by BERT's representation from the injection layer $\mathbf{H^i}$ and the queries are the injected information $\mathbf{I}$. Multihead attention requires the following weight matrices $\mathbf{W}$ and biases $\mathbf{b}$ to transform queries, keys and values (indicated by Q, K and V) and transform the attention output (indicated by O):

$$\begin{aligned}\text{params}(\text{AttentionInjection}) =& \text{params}(\mathbf{W^K}, \\ & \mathbf{W^Q}, \mathbf{W^V}, \mathbf{b^K}, \mathbf{b^Q}, \\ & \mathbf{b^V}, \mathbf{W^O}, \mathbf{b^O}) \\ =& D(2D + 2E) + 4D \\ \mathbf{W^Q}, \mathbf{W^O} \in \mathbb{R}^{D\times D}&, \mathbf{W^K}, \mathbf{W^V} \in \mathbb{R}^{E\times D}, \\ \mathbf{b^K}, \mathbf{b^Q}, \mathbf{b^V}&, \mathbf{b^O} \in \mathbb{R}^D \end{aligned} \tag{8}$$

where D indicates BERT's hidden dimension and E indicates the dimensionality of the injected embeddings. When injecting embeddings with $E = 300$ into $\text{BERT}_{\text{BASE}}$ with $D = 768$, this amounts to $\approx 1.64M$ new parameters.

Gated injection The proposed gated injection method (equations 6 to 7) only introduces the weights and biases from the projection layer, as well as the gating vector:

$$\begin{aligned}\text{params}(\text{GatedInjection}) =& \text{params}(\mathbf{W^P}, \mathbf{b^P}, \mathbf{g}) \\ =& D(E + 1) + E. \\ \mathbf{W^P} \in \mathbb{R}^{D\times E}&, \mathbf{b^P} \in \mathbb{D}, \mathbf{g} \in \mathbb{E} \end{aligned} \tag{9}$$

Gated injection of embeddings with $E = 300$ into $\text{BERT}_{\text{BASE}}$ requires $\approx 0.23M$ new parameters.

Therefore, the proposed gated injection mechanism only requires 14% of the parameters used in a multihead attention injection mechanism. Using fewer parameters results in a smaller model which is especially beneficial for injecting information during finetuning, where small learning rates and few epochs make it difficult to learn large amounts of new parameters.

H Gating Parameter Analysis

As described in section 4.1, the gating parameters g in our proposed model are initialised as a vector of zeros. During training, the model can learn to gradually inject external information by adjusting gating parameters to > 0 for adding, or < 0 for subtracting injected information along certain dimensions. Alternatively, injection stays turned off if all parameters remain at zero. Figure 2 shows a histogram of learned gating vectors for our best GiBERT models with counter-fitted (left) and dependency embedding injection (right). On most datasets, the majority of parameters have been updated to small non-zero values, letting through controlled amounts of injected information without completely overwriting BERT's internal representation. Only on Semeval B (with 4K instances the smallest of the datasets, compare section 3), more

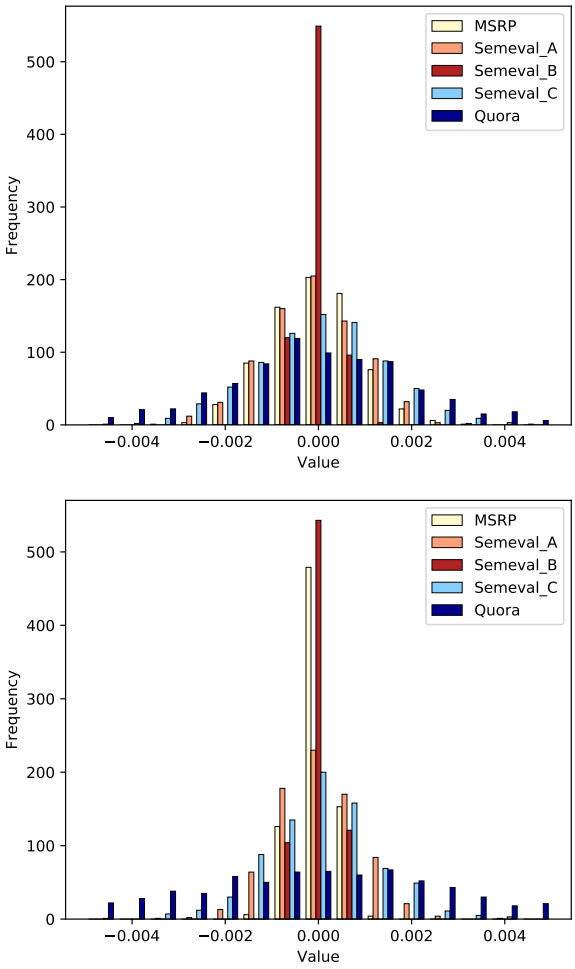

Figure 2: Histogram of the 768-dimensional gating vector g across datasets for GiBERT with counter-fitted embeddings (upper) and GiBERT with dependency embeddings (lower).

than 500 of the 768 dimensions of the injected information stay blocked out for both model variants. The gating parameters also filter out many dimensions of the dependency-based embeddings on MSRP (the second smallest dataset). This suggests that models trained on smaller datasets may benefit from slightly longer finetuning or a different gating parameter initialisation to make full use of the injected information.[5]

[5] Note that we train models for the same number of epochs, but one epoch uses all training examples contained in the dataset. This gives models trained on larger datasets more opportunity to update their parameters.