BERT Injections: Fine-Tuning BERT Using Tree-Based Word Representations to Address Syntactical Ambiguity

Stanford CS224N Custom Project

Aakriti Lakshmanan
Department of Computer Science
Stanford University
aakritil@stanford.edu

Aditya Tadimeti
Department of Computer Science
Stanford University
tadimeti@stanford.edu

Sathvik Nallamalli
Department of Computer Science
Stanford University
sathvik9@stanford.edu

Abstract

The interpretability of Large Language Models (LLMS) like BERT (Devlin et al., 2018) remains limited. Particularly, questions remain regarding how they generate predictions for a masked word when given a syntactically ambiguous sentence. Different replacements for the masked word can alter the sentence's meaning, thereby making BERT's predictions diverse. Previous research in Hewitt and Manning (2019) indicates LLMs can encode dependency parse tree information of a sentence within the hidden vector representations of each word. Hewitt's research focused on the development of a structural probe matrix, which performs a linear transformation from the squared L2 distances of the hidden vector representations to the distance between words in the parse tree. Our research extends this to determine whether BERT can use dependency parse tree information in its predictions. This would indicate BERT can leverage syntactical rules for its outputs, yielding insights for its predictions. To answer this, we perform an 'injection' on BERT by using the structural probe matrix from Hewitt and Manning (2019) to apply a transformation on BERT's hidden vectors, allowing us to 'push' them towards towards a 'gold' dependency parse tree. After this 'injection', we pass the transformed hidden vectors through the remaining layers of BERT and analyze the output probability distribution for the masked word. We conducted a qualitative analysis, doing a subjective review of the types of words the different injected models predicted for the mask, as well as a quantitative analysis, via objective, proportion-based metrics we devised using specific types of sentences. Our research supports the hypothesis that BERT can use information encoded in dependency parse trees to generate predictions that align with the added information.

1 Key Information to include

- Mentor: John Hewitt
- External Collaborators (if you have any): N/A
- Sharing project: N/A

2 Introduction

Improving the explainability of LLMs is important not only for increased transparency and trust, but to also gain a better understanding on the downstream tasks they can be used for. A responsible use of BERT necessitates a base understanding of how and why its predictions are generated. Our research therefore falls under the broader scope of improving model interpretability, as we seek to answer whether syntactical information affects the decisions and output predictions that BERT makes.

Our work is fundamental to NLP research, particularly in identifying how computers model and represent language. If we are able to gain insight regarding whether BERT utilizes dependency parse trees in its predictions, then our work sets up future research on how to explicitly incorporate different syntactical rules in LLMs.

Additional motivation for this work can be seen via vanilla BERT's predictions for a masked word in a syntactically ambiguous sentence such as the following:

They finally decided to read the books on the _____ so that they would not fail their history test.

Table 1: vanilla-BERT masked predictions & probability distribution

Predicted Words	subject	island	books	children	wall	walls	battlefield	topic	planet	school
Probabilities	0.054	0.039	0.026	0.021	0.016	0.013	0.013	0.013	0.013	0.012

The predicted words from vanilla BERT fall under multiple contexts. 'subject', 'battlefield', 'topic', and 'planet' refer to content of the books while 'wall', 'walls', and 'school' refer to physical location. If providing linguistic information via dependency parse trees to BERT influences the types of words that are predicted for the masked word, we can clearly evaluate our hypothesis that BERT is able to use linguistic information to form its predictions.

Below are two potential dependency trees for the sentence mentioned above to show examples of syntactical information that we can insert in the injected model. The circles in red highlight the difference in dependency relationships between the words, which alters the context of the sentence.

Figure 1: Parse tree for physical location syntactical representation of ambiguous sentence

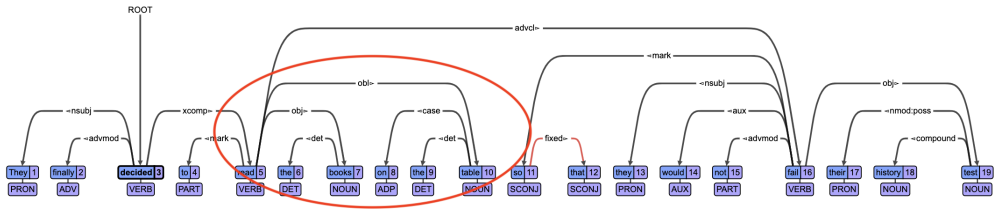
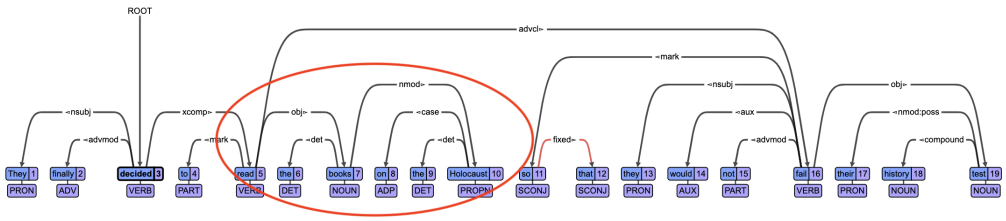


Figure 2: Parse tree for subject matter syntactical representation of ambiguous sentence



3 Related Work

Hewitt and Manning (2019): Structural Probe Previous work has been done in Hewitt and Manning (2019) to identify if BERT, which was not explicitly trained with linguistic rules, can encode linguistic information within its hidden vector representations. The premise is as follows: for a

given sentence and its corresponding parse tree, we can compute the pairwise path distance between all words, and we can also compute pairwise distance between all word vectors in the sentence. Identifying some way to relate these two distance metrics would indicate BERT could implicitly encode parse tree information within its hidden vectors.

Hewitt and Manning proposed a *structural probe*, a tool that transforms word vectors to another vector space. Under this new vector space, the distance between the transformed vectors encodes parse tree distances. This indicates that there is some linear transformation that can be done on the word vectors generated by BERT such that the distance between the updated vectors resembles the distances between the words in a parse tree. The probe, designated as B, was found by minimizing the distance, via gradient-descent, between true parse tree distances from a parsed corpus and the B-transformed distances between pairs of words in all sentences of the corpus. Specifically, B was found via the following equation:

$$\min_{B} \sum_{l} \frac{1}{|s^l|^2} \sum_{i,j} |d_{T^l}(w_i^l, w_j^l) - d_B(h_i^l, h_j^l)^2| \tag{1}$$

where l indexes each sentence in the corpus. $\frac{1}{|s^l|^2}$ is a normalization factor, where $|s^l|$ = sentence length and there are $|s^l|^2$ word pairs per sentence. $w_{1:n}^l$ represents a sequence of words passed as input to a model, indexed by i and j. $h_{1:n}^l$ represents the word vector representations produced by this model, indexed by i and j. $d_B(h_i^l, h_j^l)^2$ represents the distances between the B-transformed vectors, computed via inner product as follows: $d_B(h_i^l, h_j^l)^2 = (B(h_i^l - h_j^l))^T(B(h_i^l - h_j^l))$. $d_T(w_i^l, w_j^l)$ represents the true, human-parsed parse tree distances between pairs of words. **We describe these methods in detail as our research leverages the same structural probe in our methods.**

Hewitt and Manning's research finds that LLMs are able to encode linguistic tree structures within their vector representations. LLMs thus can somewhat replicate linguistic rules — the question of whether they directly utilize those rules is now prime for exploration.

Wahab and Sifa (2021): DiBERT This work by Wahab and Sifa attempts to train a variation of BERT, DiBERT, with dependency related information about a sentence. By training a model that correctly links each word in a sentence with it's parent, according to the spaCy Honnibal and Montani (2017) dependency tree, they show DiBERT can perform better in downstream tasks such as Natural Language Inference and Sentiment Analysis. This relates to our work as it discusses a method that uses dependency parse trees to embed information into BERT to improve its performance and awareness for this information. Our work builds on this by using the specific distances between all pairs of words in the dependency parse tree to encode syntactical information, rather than injecting a model to predict only the parent word of each word during pre-training. In addition, we work with a structural probe to move between BERT hidden vector space and syntactical spaces to improve the injection.

4 Approach

Overview The overarching task for BERT is to predict a masked word in an input sentence. The nature of how BERT functions (whether as a baseline or with different injection-modifications) as well as what the input sentence looks like varies throughout our experiments.

To see if BERT uses syntactical information about the sentence in its word predictions, we choose sentences that are syntactically ambiguous such that the content of the masked word depends on the syntactical relationship it has with the rest of the sentence. The sentences we choose offer at least two different syntactical relationships for the masked word, similar to the example in the introduction.

We then encode syntactical information for both possible dependency parse trees in the corresponding iterations of our injected model. By doing this for both possible dependency parse trees, we can see if the respective word-probability distributions are less diverse and more aligned with the specific linguistic context we provide.

Injected Model Architecture Our injection is defined as the modification and transformation of BERT's hidden vectors in accordance with the syntactical information from the dependency parse tree.

The structural probe from Hewitt and Manning (2019) was trained on the hidden vectors yielded after seven layers of BERT. For consistency, we pause the forward pass after the first seven layers after feeding an input sentence into BERT, perform the injected model to modify the hidden vectors, and resume training for the final five layers.

The input to the model is a sentence of length k. This sentence consists of the words $\{w_0, ...w_k\}$, which thereby correspond to the tokens $\{i_0, ...i_k\}$ once tokenized. Let $\mathbf{H}$ be a matrix of the hidden vectors of BERT at the seventh layer, before the injection modification, where h_i represents the hidden vector of token i. $h_i \in \mathbb{R}^{1x768}$ in accordance with BERT's specifications.

Syntactical information is represented by the distances between the tokens $\{i_0, ...i_k\}$ in the dependency parse tree for a given sentence. Refer to Figure 1 and 2 for an example. This follows from the research in Hewitt and Manning (2019) where the distance between words in the parse tree defines the syntactical structure of the sentence. From the dependency parse trees, we construct $\mathbf{D} \in \mathbb{R}^{ixi}$ to represent the distance between each pair of words according to the dependency parse tree (the path distance between each word on the tree). This matrix is denoted as the 'gold' distance matrix; further information on construction can be found in the Data section. We construct two parse trees for each representation of the sentence and calculate their associated distance matrices $\mathbf{D}^1, \mathbf{D}^2$.

Next, we utilize the structural probe B to transform the hidden vectors into vectors in the syntactical space. Once transformed, the hidden vectors now exist in a similar "syntactical" vector space as the distance matrix. We contend that modifying the hidden vectors in this new space would make changes more specific and direct, as opposed to modifying the raw $\mathbf{H}$ matrix directly. Let $\mathbf{H'}$ be the hidden vectors transformed into the syntactical space, formally defined as follows:

$$\mathbf{H'} = \mathbf{H} * B \tag{2}$$

After the transformation, we need to inject the syntactical information from the 'gold' distance matrices $\mathbf{D}^1, \mathbf{D}^2$. We do this by first calculating the squared L2 norm of the differences between all pairs of vectors in $\mathbf{H'}$. This represents the current distance between the words in the syntactical space. This matrix is denoted as $\mathbf{C}$.

$$\mathbf{C}_{i,j} = ||h'_i - h'_j||_2^2 \tag{3}$$

Now we "push" $\mathbf{C}$ to a selected distance matrix $\mathbf{D}^x$ so that we inject all of the syntactical information from the dependency parse into the hidden vectors. We do this using gradient updates. At each iteration, each hidden vector in $\mathbf{H'}$ is updated as follows where α is the learning rate:

$$\mathcal{L}(loss) = \frac{1}{i} \sum_{i,j} |\mathbf{C}_{i,j} - \mathbf{D}_{i,j}|^2 \quad , \quad h'_i = h'_i - \alpha \nabla_{\mathbf{h}_i} \mathcal{L} \tag{4}$$

Then, we take the partial gradient with respect to each h'_i in order to update the transformed hidden vectors. This loss function minimizes the difference between the squared L2 norms of the pairwise differences ($\mathbf{C}$) and the 'gold' matrix ($\mathbf{D}$). After completing the training, we need to untransform $\mathbf{H'}$ from the syntactical space back to the hidden vector space. Since B is non-invertible, we resort to using the Moore–Penrose inverse ($B^\dagger$). We do this via the following:

$$\mathbf{H'} = \mathbf{H'} * B^\dagger \tag{5}$$

From here, we feed the vectors of $\mathbf{H'}$ into the remaining five layers of BERT. Then we evaluate the output word-probability distributions as a result of this injection.

Baselines We leveraged different baselines to test the validity of our research method.

First, we use a pre-trained BERT model to give us its top ten predictions and probabilities for a masked word in a sentence — denoted as vanilla-BERT. We compare the vanilla word-probability distributions with the results of our injected variations to determine if our injections influenced BERT's predictions to align with different syntactical information we inject. vanilla-BERT lets us verify sentence ambiguity by observing if it predicts words that satisfy multiple syntactical relationships of the sentence.

Second, to determine the usefulness of B, we use an injected model that does not multiply the hidden vectors of BERT at the seventh layer (split layer) by B. Instead, the injected model still performs

gradient updates to minimize the distance between the squared L2 norm of the pairwise differences of the hidden vectors and the dependency parse tree distances ('gold' matrix). This baseline tests the theory that there exists a syntactical space under which the distance between the hidden vectors now aligns with parse tree distances.

5 Experiments

5.1 Data

Dataset Overview For our experiment, the input was syntactically ambiguous sentences, with one masked word, and two distance matrices associated with the different syntactical interpretations of the ambiguous sentence. Dataset 1 consists of six sentences sourced from Marvin and Linzen (2018), which focuses on syntactical evaluation. Dataset 2 consists of seven sentences, both manually constructed and sourced from Taraban and McClelland (1988), which focuses on comprehension/processing.

In Dataset 1, there were to different forms of each sentence tested - one with the correct form of plurality, and one with the incorrect form of plurality. For example, for the sentence "The author next to the security guards [MASK] when it is sunny," the correct plurality of the mask is singular since "author" is singular. Due to the clear distinction that plurality provides, we used Dataset 1 sentences to obtain quantitative results for our method.

Dataset 2 contained sentences with examples of noun/verb phrase attachment ambiguity and prepositional phrase attachment ambiguity. For example, in the sentence "The thieves stole all the paintings in the [MASK] while the guard slept," the masked word can attach to the noun "paintings" or the verb "stole." We used these sentences to obtain qualitative results for our method.

The output data for our experiment were the top 10 word-probability distributions of the BERT model, from the baseline BERT models and the injected BERT model.

Table 2: Dataset 1, adapted from Marvin and Linzen (2018)

The author next to the security guards [MASK] when it is sunny.
The authors that like the security guard [MASK] when it is sunny.
The mechanics said the author hurt [MASK] while working on something.
The mechanic said the authors hurt [MASK] while working on something.
The author that the security guards like injured [MASK] while working on something.
The authors that the security guard likes injured [MASK] while working on something.

Table 3: Dataset 2, adapted from Taraban and McClelland (1988) + our own sentences

The man drove the car with a broken [MASK] to the mechanic
The landlord painted all the walls with [MASK] before anyone saw
The doctor examined the patient with a [MASK] but could not determine the problem
They finally decided to read the books on the [MASK] so that they would not fail their history test
The cops scared the public with [MASK] during the parade
The band played music for animals on the [MASK] last week
The athlete trained before the dinner [MASK] so he can feel good

Data Preprocessing For every sentence, we first used spaCy to create dependency trees, which were manually adjusted using the UD Annotatrix Tyers et al. (2018) annotation tool. The CoreNLP.run tool Manning et al. (2014) was used during this process to verify manual changes to the dependency trees. After the dependency parse trees were created, they were converted to an undirected graph via NetworkX Hagberg et al. (2008), and a distance matrix was created using Floyd's algorithm to find the shortest path lengths for each pair of nodes/words in the graph.

5.2 Evaluation method

Our task was to evaluate if injecting syntactical information via the distances matrices will clarify ambiguity in BERT's predictions. We compare the output word-probability distributions between each

dependency tree under the injected model with the distribution under vanilla-BERT. Measurables for our qualitative analysis include noticing probability distributions shift for the output words between contexts and occurrence of new words native to each context.

For quantitative evaluation, we leveraged the results from Dataset 1 by calculating the frequency of 'correct' to 'incorrect' words outputted in the word-probability distributions. The 'correct' masked word for each context is based on its plurality so this heuristic allowed us to strictly define a 'correct' word. We compared the results of these metrics for each dependence parse tree during injection as well as comparing the probability distributions against vanilla-BERT.

5.3 Experimental details

Model Configurations For our vanilla-BERT baseline model, non-syntactical-probe-transformation baseline model, and injected model we used the pretrained *BertForMaskedLM* from HuggingFace with the *'bert-base-cased'* modifier. Both baseline models used the *BertTokenizer* and the injected model used *BertTokenizerFast*.

For vanilla-BERT, we allowed the input to pass through all twelve layers with no modification. For the non-syntactical-probe-transformation baseline model, we created one sub-model of vanilla-BERT by only using the first seven layers. Then we performed gradient updates directly on the hidden vectors, and ran the vectors through the rest of vanilla-BERT.

Our injected model follows a similar approach, but transforms the hidden vectors prior to the gradient updates. We ran gradient updates until loss reached a convergence value, which was set at $0.03 \cdot initialLoss$ where $initialLoss$ is the loss prior to performing gradient updates. Before settling on Mean Square Error as our loss, we experimented with custom loss functions based on the raw sum of differences between the two matrices, Mean Absolute Error, etc.

For the learning rate during the gradient updates, we experimented with: 0.1, 0.01, 0.001, and 0.0001. The final learn rate we chose is 0.001 as it ensures that our loss continues to decrease and we converge to an appropriate minimum. Prior to this, we experimented with using a scheduled learn rate, which adjusts the learn rate dynamically. We also experimented with setting convergence to be based on the norm of the gradient, loss to be a fixed value (i.e. 0.009), etc. These methods proved to be ineffective as they either caused the training time to be too long or weren't applicable for all kinds of sentences.

Since all the sentences differ in construction, loss, and distance matrices, we settled on a more consistent method of loss convergence. A visual representation of the effects of the model injection on the hidden vectors can be found in the appendix in the form of heatmaps.

5.4 Results

For the Marvin/Linzen dataset, the results of our experiment are summarized in the following table. The counts are sourced from the top ten predictions of the injected model. As seen in the last few rows of Table 4, we see similar results in terms of counts. However, a clear distinction can be seen when we examine the prediction probabilities of the model, as shown in Table 5. A value of "N/A" is given to words not found in the top ten predictions.

Table 4: Counts of Plural Words vs Singular Words

	Correct Tree		Incorrect Tree	
Sentence	# Correct	# Incorrect	# Correct	# Incorrect
The author next to the security guards [MASK] when it is sunny.	3	0	1	0
The authors that like the security guard [MASK] when it is sunny.	2	1	0	8
The mechanics said the author hurt [MASK] while working on something.	2	1	2	1
The mechanic said the authors hurt [MASK] while working on something.	1	1	1	1
The author that the security guards like injured [MASK] while working on something.	1	1	1	1

When compared to the baseline probabilities, the injected plural tree model led to an average 0.012 increase in probabilities for "themselves". Similarly, the injected singular tree model to an average 0.39 increase in probabilities for "himself." We also see the adverse relationship represented in the

Table 5: Probabilities of Plural vs Singular Words

	Plural Tree Probabilities		Singular Tree Probabilities		Vanilla BERT Probabilities	
Sentence	"themselves"	"himself"	"themselves"	"himself"	"themselves"	"himself"
The mechanics said the author hurt [MASK] while working on something.	0.048	0.806	0.018	0.864	0.011	0.858
The mechanic said the authors hurt [MASK] while working on something.	0.447	0.073	0.0812	0.108	0.584	0.106
The author that the security guards like injured [MASK] while working on something.	0.143	0.024	N/A	0.080	0.00088235	0.00044496
The authors that the security guard likes injured [MASK] while working on something.	N/A	0.012	N/A	0.069	0.00034945	0.00056551

data. The injected plural tree model led to an average 0.012 decrease in probabilities for "himself," while the injected singular tree model led to an average .12 decrease in probabilities for "themselves."

For Dataset 2, we evaluate the sentences across both baselines and the injected model by comparing the top 10 word-probability distributions.

Table 6: vanilla-BERT Baseline Predictions, View full table in Appendix

The man drove the car with a broken [MASK] to the mechanic	leg 0.200, arm 0.134 windshield 0.083, wheel 0.077 nose 0.074, tire 0.045 window 0.033, neck 0.033 engine 0.027, head 0.015
They finally decided to read the books on the [MASK] so that they would not fail their history test	subject 0.0543, island 0.039 books 0.026, children 0.021 wall 0.016, walls 0.013 battlefield 0.013, topic 0.013 planet 0.013, school 0.012
The doctor examined the patient with a _____ but could not determine the problem	mirror 0.123, needle 0.102 doctor 0.065, physician 0.058 flashlight 0.054, knife 0.034 telescope 0.033, bandage 0.016 blanket 0.016, lens 0.0145

Across all sentences, we notice that the no-transformation baseline has an almost identical collection of words and order between the two dependency parse trees, in some cases the probability distributions for a specific word vary only by 0.001-0.002. Compared to the vanilla-BERT baseline, we still see a high collection of words that are ambiguous to both contexts of the sentence in no-transformation. For the injected model, at the minimum, we see clear re-orderings of word-probability distributions and for select sentences, we see new words in the distribution that fit with the inputted dependency parse tree. See Table 6 and 7.

6 Analysis

In order to examine the effects of the injected model on the outputs of BERT, we employed two different methods of evaluation - quantitative evaluation for Dataset 1, and qualitative evaluation for Dataset 2. Quantitative evaluation of Dataset 1 included raw counts of different word plurality as well as analysis of the prediction probabilities themselves.

We realized that the results of our injected model were dependent on the outcomes of the vanilla-BERT baseline model. In some cases, the vanilla-BERT model provided predictions for both interpretations of a sentence. For example, for the sentence "The man drove the car with a broken [MASK] to the mechanic," vanilla-BERT provides results such as {leg, arm, nose} but also provides results such as {tire, window, engine}. When the different parse trees were provided to the injected model, we are

Table 7: no-syntactical-transformation BERT Baseline and injected BERT Predictions.**D1** represents the dependency tree encoded with the first context, same for **D2**. View full table in Appendix.

Sentence	no-transformation D1	no-transformation D2	injected D1	injected D2
The man drove the car with a broken [MASK] to the mechanic	**[MASK] -> man** wheel 0.148, leg 0.096 head 0.074, arm 0.057 car 0.037, window 0.030 chair 0.029, neck 0.022 body 0.022, man 0.019	**[MASK] -> car** wheel 0.143, leg 0.109 head 0.075, arm 0.063 car 0.033, window 0.030 chair 0.026, neck 0.024 body 0.022, nose 0.018	**[MASK] -> man** arm 0.272, leg 0.123 hand 0.071, stick 0.039 stone 0.029, bone 0.021 finger 0.010, glass 0.010 chain 0.010, hammer 0.010	**[MASK] -> car** note 0.062, hand 0.058 arm 0.054, word 0.049 light 0.048, metal 0.028 silver 0.019, glass 0.014 broken 0.012, gold 0.011
They finally decided to read the books on the [MASK] so that they would not fail their history test	**[MASK] -> books** books 0.040, children 0.017 book 0.016, Bible 0.0124 island 0.011, subject 0.010 map 0.010, truth 0.009 wall 0.008, table 0.007	**[MASK] -> read** books 0.038, children 0.016 book 0.015, Bible 0.013 subject 0.0125, island 0.011 map 0.010, truth 0.009 wall 0.009, table 0.007	**[MASK] -> read** ground 0.043, wall 0.041 floor 0.030, table 0.026 back 0.019, top 0.017 river 0.015, side 0.013 throne 0.013, scroll 0.012	**[MASK] -> read** wall 0.045, ground 0.039 children 0.034, place 0.032 back 0.023, top 0.015 throne 0.014, city 0.014 library 0.013, street 0.013
The doctor examined the patient with a _____ but could not determine the problem	**[MASK] -> patient** physician 0.057, doctor 0.049 patient 0.038, mirror 0.031 doubt 0.025, diagnosis 0.020 skull 0.013, psychiatrist 0.012 needle 0.011, surgeon 0.011	**[MASK] -> doctor** physician 0.059, doctor 0.050 patient 0.036, mirror 0.034 doubt 0.025, diagnosis 0.020 psychiatrist 0.013, skull 0.012 needle 0.012, surgeon 0.011	**[MASK] -> patient** cross 0.076, torch 0.032 stone 0.029, stick 0.025 certainty 0.024, hammer 0.016 knife 0.014, horse 0.012 curse 0.012, headache 0.011	**[MASK] -> doctor** cross 0.057, torch 0.030 stone 0.026, hammer 0.021 certainty 0.018, curse 0.018 stick 0.017, headache 0.016 horse 0.016, spear 0.015

able to push the model's predictions in both directions, providing {arm, leg, hand, body, bone, foot} for the context where broken is attached to "man," and providing {gold, light, glass, metal, silver} for the context where broken is attached to "car." Since the initial vanilla-BERT model was able to recognize both interpretations initially, we were able to widen the distinction via the injected model. In cases where vanilla-BERT only recognized one interpretation, we often had unrelated words or punctuation provided as predictions for the unrecognized dependency parse.

In other words, we see that BERT is able to recognize some syntactical information via the dependency trees, but not all. When BERT is unable to initially recognize the syntactical information, it does not use it while predicting, and attempts to push BERT in this direction do not lead to promising results.

Between vanilla-BERT and no-syntactical-transformation, since we do not use B, the loss between the squared L2 norms of the difference of the hidden vectors and 'gold' matrix is incredibly high (ex: 317529.9256). Despite the gradient updates, since the vectors are not in the syntactical space, the outputs of no-syntactical-transformation remain ambiguous and similar to vanilla-BERT. The occurrence of new words such as {light, chain, gold} in sentence 1 and {library, street} in sentence 2 in the injected model shows promise of our method as these words are native to the injected context and could indicate that the syntax was clarified to BERT.

The question of whether or not BERT utilizes linguistic context when making predictions was partially answered by the Marvin and Linzen dataset. As seen in Table 5, the plural and singular dependency trees push the probabilities of vanilla-BERT in the proper direction as initially hypothesized. As mentioned above, this distinction was clear due to the fact that vanilla-BERT provided both versions as possible predictions prior to any injections.

7 Conclusion

Main Findings From the results of the Marvin dataset, we learned that BERT does indeed use the injected syntactical information in its predictions. By using the input dependency tree, our injection allowed BERT to predict words that matched the plurality of its dependent word. Through a comparison of our vanilla-BERT baseline and no-syntactical-transformation baseline, we concluded that the probe B is necessary and contributes significantly to the injection. Though with slightly different probability distributions, words in the no-syntactical-transformation baseline were nearly identical to the vanilla-BERT baseline, which supports the idea that gradient updates without B are insignificant. Dataset 2 also revealed that the success of the injected model is dependent on the initial outputs of vanilla-BERT. In some sentences, we noticed the introduction of new words that

fit the corresponding context of the dependency tree, not found in vanilla-BERT or no-syntactical-transformation, occur in our injected model. This observation, coupled with the shifts in word-probability distributions between the dependency trees in our injected model, support our hypothesis that this injection has clarified ambiguity. However, we noticed marginal changes from our injection between different dependency trees for some sentences. These takeaways indicate sizeable potential for improvement on our method to apply to more types of sentences. Given that sentences are highly diverse in structure, length, and dependencies, it was difficult to develop a method that delivered consistent results across all sentences. We tested multiple model configurations and found that a learning rate of 0.001 and defining convergence as reaching $0.03 \cdot initialLoss$ provided the best results. This model configuration can also be applied to different use cases, such as debiasing. Overall, with reasonable room for improvement, we added to BERT's interpretability and understanding of functionality by showing changes in results due to our injection.

Limitations Since the structural probe matrix *B* provided in Hewitt and Manning (2019) was trained on layer seven of BERT, we were limited to performing our injection at layer seven as well. In the future, we would like to train *B* at each layer of BERT-base to investigate how performing the injection at a different layer can contribute to BERT's ability to use syntactical information in its predictions. As our task involves ambiguous sentences for the input and distinct dependency parse trees for each sentence, our dataset was constrained in size. There was no publicly available dataset for this purpose, so more time was devoted to developing a data pipeline and gathering sentences that could be potentially used.

Future Work We would like to expand the dataset to contain more sentences where vanilla-BERT can recognize both interpretations of the sentence. In addition, we are interested in seeing the impacts of such an injection on other LLMs. As we mentioned the increased usage of LLMs in downstream tasks, we would like to evaluate how well our method works for tasks other than masked word prediction, such as sentiment analysis and debiasing.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv.

Aric Hagberg, Pieter Swart, and Daniel S Chult. 2008. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *The Stanford Natural Language Processing Group*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

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

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models.

Roman Taraban and James L McClelland. 1988. Constituent attachment and thematic role assignment in sentence processing: Influences of content-based expectations. *Journal of Memory and Language*, 27(6):597–632.

Francis M. Tyers, Mariya Sheyanova, and Jonathan North Washington. 2018. Ud annotatrix: An annotation tool for universal dependencies. In *Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories (TLT16)*, pages 10–17.

Abdul Wahab and Rafet Sifa. 2021. Dibert: Dependency injected bidirectional encoder representations from transformers. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8.

A Appendix (optional)

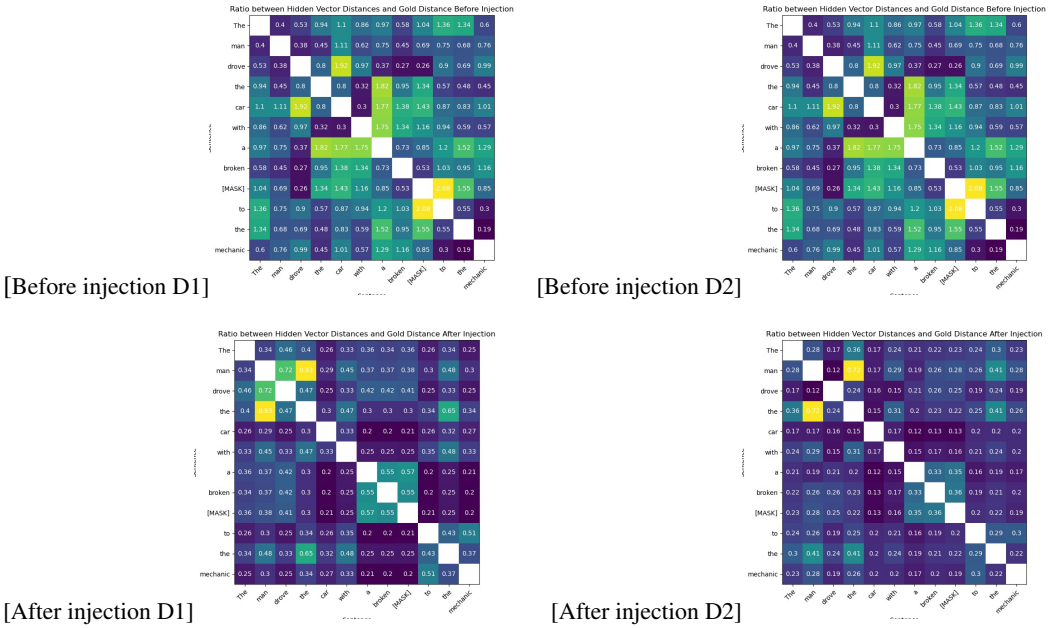
In this Appendix, we have included the full, expanded versions of some of the tables we truncated in the earlier parts of this paper. We also included some extra visualizations and figures here.

Table 6: **Full Table** vanilla-BERT Baseline Predictions

Sentence	word-probability distributions
They finally decided to read the books on the [MASK] so that they would not fail their history test	subject 0.0543, island 0.039 books 0.026, children 0.021 wall 0.016, walls 0.013 battlefield 0.013, topic 0.013 planet 0.013, school 0.012
The man drove the car with a broken [MASK] to the mechanic	leg 0.200, arm 0.134 windshield 0.083, wheel 0.077 nose 0.074, tire 0.045 window 0.033, neck 0.033 engine 0.027, head 0.015
The band played music for animals on the [MASK] last week	road 0.074, show 0.053 radio 0.050, beach 0.044 streets 0.044, farm 0.033 tour 0.026, street 0.0216 train 0.019, stage 0.018
The athlete trained before the dinner in the [MASK] so he can feel good	gym 0.163, restaurant 0.068 kitchen 0.057, cafeteria 0.046 park 0.039, courtyard 0.021 morning 0.020, lobby 0.020 hotel 0.020, bathroom 0.018
The cops scared the public with [MASK] during the parade	guns 0.108, fear 0.078 them 0.068, bullets 0.060 gunfire 0.042, violence 0.035 it 0.034, questions 0.027 threats 0.026, terror 0.024
The doctor examined the patient with a _____ but could not determine the problem	mirror 0.123, needle 0.102 doctor 0.065, physician 0.058 flashlight 0.054, knife 0.034 telescope 0.033, bandage 0.016 blanket 0.016, lens 0.0145
The landlord painted all the walls with _____ before anyone saw	graffiti 0.164, flowers 0.134 paint 0.101, blood 0.043 it 0.035, roses 0.033 them 0.023, murals 0.020 red 0.020, gold 0.018

Table 7: **Full Table** no-transformation BERT and injected BERT Predictions

Sentence	no-transformation D1	no-transformation D2	injected D1	injected D2
They finally decided to read the books on the [MASK] so that they would not fail their history test	**[MASK] -> books** books 0.040, children 0.017 book 0.016, Bible 0.0124 island 0.011, subject 0.010 map 0.010, truth 0.009 wall 0.008, table 0.007	**[MASK] -> read** books 0.038, children 0.016 book 0.015, Bible 0.013 subject 0.0125, island 0.011 map 0.010, truth 0.009 wall 0.009, table 0.007	**[MASK] -> books** ground 0.043, wall 0.041 floor 0.030, table 0.026 back 0.019, top 0.017 river 0.015, side 0.013 throne 0.013, scroll 0.012	**[MASK] -> read** wall 0.045, ground 0.039 children 0.034, place 0.032 back 0.023, top 0.015 throne 0.014, city 0.014 library 0.013, street 0.013
The man drove the car with a broken [MASK] to the mechanic	**[MASK] -> man** wheel 0.148, leg 0.096 head 0.074, arm 0.057 car 0.037, window 0.030 chair 0.029, neck 0.022 body 0.022, man 0.019	**[MASK] -> car** wheel 0.143, leg 0.109 head 0.075, arm 0.063 car 0.033, window 0.030 chair 0.026, neck 0.024 body 0.022, nose 0.018	**[MASK] -> man** arm 0.272, leg 0.123 hand 0.071, stick 0.039 stone 0.029, bone 0.021 finger 0.010, glass 0.010 chain 0.010, hammer 0.010	**[MASK] -> car** note 0.062, hand 0.058 arm 0.054, word 0.049 light 0.048, metal 0.028 silver 0.019, glass 0.014 broken 0.012, gold 0.011
The band played music for animals on the [MASK] last week	**[MASK] -> band** show 0.168, festival 0.046 album 0.025, tour 0.021 night 0.020, program 0.019 bill 0.0159, concert 0.014 stage 0.013, song 0.013	**[MASK] -> animals** show 0.167, festival 0.045 album 0.025, tour 0.022 program 0.019, night 0.019 bill 0.016, concert 0.014 stage 0.013, song 0.012	**[MASK] -> band** streets 0.505, street 0.094 road 0.026, earth 0.015 ground 0.014, internet 0.013 Internet 0.013, planet 0.012 roads 0.012, beach 0.012	**[MASK] -> animals** streets 0.353, street 0.096 road 0.040, ground 0.020 playground 0.017, beach 0.016 roads 0.013, grid 0.013 roof 0.013, pavement 0.013
The athlete trained before the dinner in the [MASK] so he can feel good	**[MASK] -> dinner** gym 0.116, morning 0.055 restaurant 0.040, kitchen 0.030 evening 0.022 park 0.021 stadium 0.020, cafeteria 0.019 house 0.018, hospital 0.018	**[MASK] -> athlete** gym 0.114, morning 0.056 restaurant 0.041, kitchen 0.030 park 0.022, evening 0.022 stadium 0.020, cafeteria 0.020 house 0.018, hospital 0.018	**[MASK] -> dinner** world 0.020, morning 0.020 field 0.019, sky 0.016 room 0.015, park 0.013 ring 0.011, yard 0.011 west 0.011, street 0.010	**[MASK] -> athlete** morning 0.032, world 0.020 sky 0.018, day 0.013 field 0.012, ring 0.011 room 0.011, arena 0.010 end 0.009, west 0.009
The cops scared the public with [MASK] during the parade	**[MASK] -> public** it 0.128, them 0.090 violence 0.067, fireworks 0.054 police 0.021, him 0.015 cars 0.015, riot 0.013 riots 0.012, protests 0.012	**[MASK] -> cops** it 0.131, them 0.089 violence 0.069, fireworks 0.058 police 0.020, cars 0.016 him 0.0144, riot 0.013 fear 0.012, riots 0.012	**[MASK] -> public** joy 0.284, laughter 0.032 fear 0.03, fireworks 0.021 excitement 0.018, words 0.014 hope 0.013, support 0.011 Twitter 0.01, money 0.008	**[MASK] -> cops** fear 0.117, them 0.047 joy 0.025, excitement 0.017 dust 0.016, concern 0.016 help 0.013, hope 0.012 alarm 0.012, laughter 0.010
The doctor examined the patient with a _____ but could not determine the problem	**[MASK] -> patient** physician 0.057, doctor 0.049 patient 0.038, mirror 0.031 doubt 0.025, diagnosis 0.020 skull 0.013, psychiatrist 0.012 needle 0.011, surgeon 0.011	**[MASK] -> doctor** physician 0.059, doctor 0.050 patient 0.036, mirror 0.034 doubt 0.025, diagnosis 0.020 psychiatrist 0.013, skull 0.012 needle 0.012, surgeon 0.011	**[MASK] -> patient** cross 0.076, torch 0.032 stone 0.029, stick 0.025 certainty 0.024, hammer 0.016 knife 0.014, horse 0.012 curse 0.012, headache 0.011	**[MASK] -> doctor** cross 0.057, torch 0.030 stone 0.026, hammer 0.021 certainty 0.018, curse 0.018 stick 0.017, headache 0.016 horse 0.016, spear 0.015
The landlord painted all the walls with _____ before anyone saw	**[MASK] -> paint** them 0.151, it 0.109 graffiti 0.066, paint 0.042 pictures 0.034, him 0.028 flowers 0.022, blood 0.014 her 0.011, water 0.011	**[MASK] -> walls** them 0.145, it 0.104 graffiti 0.072, paint 0.045 pictures 0.035, him 0.027 flowers 0.024, blood 0.015 her 0.011, water 0.011	**[MASK] -> paint** graffiti 0.066, mud 0.046 paint 0.044, him 0.033 dust 0.032, gold 0.031 red 0.026, blood 0.025 it 0.022, dirt 0.019	**[MASK] -> walls** dust 0.066, red 0.048 paint 0.041, graffiti 0.041 mud 0.040, gold 0.036 dirt 0.031, blood 0.025 yellow 0.023, stones 0.020



[Before injection D1] [Before injection D2]

[After injection D1] [After injection D2]

Figure 3: Ratio between hidden vector distances and gold distance matrix

Figure 4: Example of loss declining to convergence for both dependency trees of a sentence