Role of BERT in misinformation detection

Anonymous Authors[1]

1. Abstract

This work revolves around the role of BERT and attention in the problem of information verification. Given a claim, it is verified against an annotated corpus through evidence sentences labelled as SUPPORTS, REFUTES and NOT ENOUGH INFO. The work of (Zhou et al., 2019) shows the performance of evidence aggregation through attention on graph neural networks for the task. Later that year, (Knyazev et al., 2019) explained the conditions under which optimal generalization performance is achieved while using attention over graph neural networks. The work of (Clark et al., 2019) brought to notice the unrewarding scattering effect of the dot-product attention employed thus far towards attention tasks. In the next year, (Liu et al., 2020) introduces kernel attention and shows better results along with more explainability via employing methods from (Knyazev et al., 2019) and kernel-based attention over graph neural networks. Before (Liu et al., 2020), all work in this direction employing attention (e.g. (Zhou et al., 2019)) made use of BERT. It only makes sense to try and attempt to understand the merits and recently noted demerits of the architecture as has been explained in (Jawahar et al., 2019) and (Clark et al., 2019). By addressing the latest issues, (Liu et al., 2020) has introduced an architecture that produces state-of-the-art result in misinformation classification. This work is thus an attempt to understand the role played by different components in that architecture.

2. Review and Critique

2.1. Introduction

Remember the activities of Cambridge Analytica that are rumoured to have played a key role in 2016 US elections, Brexit and other major political events. Apparently, it developed algorithms that analyzed and consequently learnt to influence social paradigms through selective and targeted abridged information spreading on social networking sites. To combat digital mishaps like that, it is paramount that wrong information be verified and stopped from spreading. Misinformation detection has gained increasing importance in the light of recent events including misinformation spreading regarding the novel coronavirus COVID-19. Several algorithmic architectures have been proposed towards the same. At the center of it all is attention. Graph attention networks is an intuitive way of modelling the relation between claims and the related documents. Given a claim, a system first retrieves related sentences from the corpus, develops a graph out of the claim-retrieved sentence pair as nodes, carries out joint reasoning on these sentences and marks each sentence as SUPPORTS, REFUTES and NOT ENOUGH INFO against the claim. Joint reasoning is conducted over the sentences as the corpora available may also be noisy for example, fabricated but approved article taken from the internet. A sentence picked from a fabricated article may give a different label when put to reasoning independently as compared to jointly with all the other related sentences. A key component common to all the proposed attention based language models is the BERT architecture. It stands for Bidirectional Encoder Representations from Transformers and has done tremendously well in problems involving language processing. In this work, our focus will be two-fold - on comprehending the work by (Clark et al., 2019) that gives excellent justifications of how BERT is able to perform so well, and secondly, on exploring the novelty introduced in (Liu et al., 2020) towards misinformation detection.

2.2. Review of works in this direction

The work in (Zhou et al., 2019) is guided by the fact that a graph-based evidence aggregation will allow better interaction between each evidence for a particular claim. Specifically, they propose a graph-based evidence aggregating and reasoning (GEAR) framework based on a fully connected evidence graph. The model further uses BERT to understand the semantic structure. To understand why such an interaction between evidence is necessary, consider the following example from the paper.

Claim: Giada at Home was only available on DVD.

Evidence 1: Giada at Home is a television show and first aired on October 18, 2008, on the Food Network.

Evidence 2: Food Network is an American basic cable and satellite television channel.

Notice that the claim could have been labelled as NOT ENOUGH INFO based only on Evidence 1. Only after facts from Evidence 1 and 2 and reasoned over together, the claim

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

can be correctly labelled as REFUTES.

The GEAR method is three-step pipeline - 1. Document retrieval 2. Evidence selection 3. Claim verification based on evidences.

For the first step, a constituency parser is used on the claim to develop potential entities. This entities are used as search queries for the Online MediaWiki API to find relevant Wikipedia documents. A constituency parser basically refers to developing a tree structure based on syntactic relations of the sentence. In this work, it is done through the one provided by AllenNLP. The constituents identified as potential entities are queried using the API and the top 7 results are stored as candidate article set.

The second step is of evidence selection which refers to picking relevant sentences from the articles selected in the last step. This is done by calculating a relevance score between the article sentences and the claim and the top 5 are selected as the final evidence. Further, during training of the model, depending on the validation set result a threshold τ is used to eliminate sentences that have a relevance score less than τ. The optimizing function used for training is,

$$\sum \max(0, 1 + s_n - s_p) \tag{1}$$

where s_p ans s_n are the relevance scores of positive and negative samples respectively.

Next step is the last step of labelling the claim with respect to the selected evidence. This step introduces novelty to the procedure. First, for each claim - evidence sentence pair (e_i, c), an embedding is developed using BERT. The augmentation of the claim and evidence into one node is guided by the belief that for efficient message traversal across the other nodes in the graph, information from the claim is required along with that from the corresponding evidence. An embedding is developed for claim c as well. Now once the nodes are developed, reasoning needs to be propagated among the evidences. That will translate to message passing on the evidence-claim graph developed previously. As the process begins, the hidden states are represented as, $\mathbf{h}^t = \mathbf{h}_1^t, ..., \mathbf{h}_N^t$ where N is the number of evidences selected for the claim, c, $\mathbf{h}_i^t \in \mathcal{R}^{F\times 1}$, F is the number of features in each node, t is the layer. Next comes the evidence reasoning network (ERNet) to propagate information on the network. The attention coefficient between neighboring nodes i and j is calculated as follows,

$$p_{i,j} = \mathbf{W}_1^{t-1}(ReLU(\mathbf{W}_0^{t-1}(h_i^{t-1}||h_j^{t-1}))) \tag{2}$$

where $\mathbf{W}_1^{t-1} \in \mathcal{R}^{1\times H}$ and $\mathbf{W}_0^{t-1} \in \mathcal{R}^{H\times 2F}$ are weight matrices. Then a softmax normalization is applied on $p_{i,j}$ to get the normalized attention coefficients, $\alpha_{i,j}$. A linear combination of $\alpha_{i,j}$ with the previous hidden layer representation of jth neighbour, h_j^{t-1}, gives the current hidden layer representation of node i, h_i^t. After $t = 0, 1, ..., T$ layers of ERNet, the final hidden states $h_i^T, i = 1, ..., N$ are sent to evidence aggregator to determine the final label. The evidence aggregator is basically an operation on all the final hidden states obtained so far. Three operations have been tried in the paper - attention, mean and max. The attention operation works similarly as described in equation (2) with input as the concatenation of c and h_j^T. The max aggregator carries out element wise maximum operation on all the hidden states. The mean aggregator carries out element wise mean operation on all of the hidden states. The final prediction is given by using a softmax on the linear combination of the outputs of the aggregating operation.

The work in (Knyazev et al., 2019) analyzes the generalization efficiency of employing attention over nodes in graph neural networks. Note than, in practice, attention can be employed over nodes as well as edges. Here, the focus will be on node attention. This paper draws a similarity between attention in convolutional neural networks (CNN) and pooling in graph neural network (GCN). Attention in a CNN can be expressed as

$$Z = \alpha \bigodot X \tag{3}$$

$X \in \mathcal{R}^{\mathcal{N}\times\mathcal{C}}$ is the C-dimensional input and $\bigodot$ is element wise product. While pooling in a CNN happens over well-defined grids, in a GCN, the pooling region is defined by a cluster of similar nodes. But in top-k pooling, only a fraction of input nodes is selected and the pooling operation is performed on it. But that is equivalent to

$$Z_i = \begin{cases} \alpha_i X_i & \forall x \in P, \\ \varnothing & \text{otherwise} \end{cases}$$

where P is the set of indices selected for pooling. The only difference between the above equation and (3) is the dimension of the output Z_i. Thus, for a ratio, $r = \frac{|P|}{N}$, pooling in GCN is same as attention in CNN. This observation inspired creation of an unified block of attention and pooling in GCN. This block is analysed under 3 different tasks that enables studying the power of this block to generalize better.

The paper analyses the block with graph convolution network (GCN) as well as Graph Isomorphisms network (GIN) but here we will only focus on the results pertaining to GCN as that is the component that will be required later. For the model, a ratio of $r = 0.8$ was selected and $\tilde{\alpha}$ was selected as the threshold for selecting nodes to be operated with the pool operation. An interesting feature that was noted here was that the model assigned similar value of $\tilde{\alpha}$ to local nodes. Now to train the model that predicts these coefficients for nodes, methods used are LinearProjection and DiffPool. LinearProjection is a where α is the single layer projection of X and for DiffPool a separate GCN is trained. Both cases use a softmax activation in the final layer and training loss is defined by the Kullback-Liebler divergence loss.

The main observations of the paper underscore that using attention over nodes in GCN can help them generalize better. The three key factors influencing the generalization ability of GCN is the initialization, strength and the hyperparameters of the GCN and the attention model. Further in this regard, in the supplementary material, the authors explore to applying attention model to which layer leads to the optimal result. They compare applying attention to the initial layers and the deeper layers and find that while it is better for the overall performance to have attention on the deeper layers, it is both computationally efficient and easier to interpret to have attention on the lower layers as it translates to selecting what features are physically important and reduces the input size for the higher layers.

The work in (Jawahar et al., 2019) and (Clark et al., 2019) is targeted towards understanding the working of BERT. While (Jawahar et al., 2019) aims to analyse the representations learnt in different layers of BERT, the efforts in (Clark et al., 2019) are geared towards understanding the attention mechanism in BERT.

(Jawahar et al., 2019) categorises their work into 3 levels of information - phrase level, probing tasks to show the hierarchy of information captured in BERT and the capacity of BERT to keep track of associated subjects and verb with multiple such pairs. For phrasal representation, the span of each layer was visualized in the low-dimension and it was seen that ”chunks” of similar phrases were better represented in the graph of lower layers as compared to higher layers. This proved that the lower layers are better than the higher ones at capturing phrasal relations. Next, for the understanding the hierarchy of linguistic features learnt by BERT, probing methods are used. Probing method here refers to feeding the output of a model for an auxiliary classification task. If the desired feature can the be predicted well from that output, then it is highly likely that the model that generated it, also encodes that property well. Equivalently, here it will be used to assess the ability of each layer to model certain linguistic features. Their experiments show that BERT encodes surface information at the bottom, syntactic information in the middle, semantic information at the top. Further, it was found that the the higher layers outperforms the trained version in the task of predicting sentence length. For the third task of tracking the relations between the nouns and verb present in a sentence, it was found that the deeper layers were better at encoding the relations between the nouns and the corresponding verbs in each sentence. Lastly, to understand how BERT learns the compositional structure, the authors use Tensor Product Decomposition Networks (TPDN) which explicitly composes the representation of the input token based on the predetermined role scheme. A role scheme for a word, for example, can be based on the path from the root node to itself in the syntax tree. It is believed that if a TPDN can approximate the representation learned by a neural model, then it is highly likely that the predetermined role scheme is a good indication of the compositional structure learnt by the model.

(Clark et al., 2019) attempts to study the ”attention-maps” in BERT. Specifically, the first task they probe is the general pattern of attention in attention heads. For this the attention maps were extracted from BERT-base over 992 Wikipedia segments. Their findings include the fact that most attention heads either focus on the previous token or on the next token and very less attention is given to the current token. They also found more attention was given to the [CLS] and [SEP] tokens and have provided arguments as to why it is not just a statistical consequence. The testing shows that the increased attention on these special tokens is like the default operation like ”no-op” of the attention head as in when the attention head’s function is not applicable. This hypothesis is validated by one of their experiments when the gradient of the loss function with respect to attention on [SEP] token was tracked and was found to be low. This would translate as the loss does not depend much on the attention spent on [SEP] and therefore validating the attention on [SEP] as ”no-op” option. Further, and importantly, they find that some lower layer attention heads have a scattered attention in the sense that they spend at most 10% of their attention mass on any single word. The term used in the paper is attention entropy where a high entropy means highly distracted and less attention. They have also noted that the last layer has a very broad attention for the [CLS] token which turns out to be fruitful in tasks where an aggregate representation of the whole input is required.

(Clark et al., 2019) further attempts to understand the feature of language learnt by each attention head. An important step here was to modify the architecture of BERT to adapt to word-word attention maps instead of token-token attention maps. The first feature evaluated in this regard is the dependency relation,as in how well a particular attention head captures semantic dependencies. Although the results are inconclusive, some attention heads showed good results on particular relations. Next, they aim to verify the attention maps in combined attention heads. The important observations found here are that the English syntax is pretty well captured in BERT’s attention maps and that the vector representations captures equivalent syntactic information as the attention maps. The next point of probing is whether attention heads behave more similar to some attention heads compared to others. This is measured by using the Jensen-Shannon Divergence (JSD) between any two attention distributions and for visualizing, two dimensional embedding are developed such that the Cartesian distance is a approximates the JSD well. It was found that heads in the same layer often behave similar.

The work in (Liu et al., 2020) introduces the Kernel Graph Attention Network (KGAT) to address the fact verification problem as in (Zhou et al., 2019) but takes a different approach considering all the problems and advancements highlighted in the papers discusses above. In this section, I will discuss the broad details of KGAT. The next section will compare, contrast and draw a relation between the papers discussed.

The paper starts with describing their kernel attention mechanism but the procedure to select articles and sentences that act as evidence come first in the procedure, so I will be discussing that first. In (Liu et al., 2020), the procedure to select articles uses a constituency parser on the given claim to extract potential entities. Then the online MediaWiki API is used to extract documents that have a match with these entities.The sentences that serve as evidence are further extracted from these retrieved documents using the ESIM and BERT based sentence retrieval methods. In this work we will focus on the BERT method of sentence retrieval where the hidden state of the [CLS] token in each claim-evidence pair node is projected to a score and the pairwise loss is used to train the model.

Next comes the task of verifying the claim against the evidence sentences retrieved in the last step. This is where the novelty in the architecture of KGAT lies, in introducing edge kernels for information propagation across the graph and node kernels to reason over each evidence. The graph is a complete graph (ever pair of node is connected by an edge) made of the claim and the retrieved evidence sentences as nodes. Specifically, for each claim c, with l evidence sentences, the graph, G, is constructed with nodes $N = n^1, .., n^p, ., n^l$. The label, y, is predicted as follows,

$$P(y|G) = \sum_{p=1}^{l} P(y|n^p, G)P(n^p|G) \tag{4}$$

The term $P(y|n^p, G)$ calculates the label with respect to all the evidence aggregated from the graph while $P(n^p|G)$ denotes the p^{th} evidence selection probability which translates as a measure of how strong a particular evidence is for its corresponding claim. When combined with $P(y|n^p, G)$, it basically weighs the labels with respect to each evidence sentence.

The procedure starts with developing the initial node representations using a pre-trained BERT model. Each node n^p is composed of m tokens of claim and n tokens of evidence both including [SEP]. The hidden representations, H^p, are obtained using BERT and the representation, H^p_0, of the first token, [CLS], is called the first representation of z^p. For further developing the representations using attention with respect to the other nodes, the edge kernel uses token-level attention. The attention coefficient, $\alpha_i^{p\to q}$, for the i^{th} token in q due to p, is calculated as,

$$\alpha_i^{p\to q} = softmax_i(Linear(\vec{K}(M_i^{p\to q}))) \tag{5}$$

where $i = 1, ..., K$ and each element of $\vec{K}$ is the Gaussian kernel defined as follows,

$$K_k(M_i^{p\to q}) = \log \sum_j \exp(-\frac{(M_{ij}^{p\to q} - \mu_k)^2}{2\delta_k^2}) \tag{6}$$

where μ_k and δ_k are the mean and width of the k^{th} kernel for $k = 1, ..., K$ and

$$M_{ij}^{p\to q} = \cos(H_i^p \to H_j^q) \tag{7}$$

which translates as the similarity between the BERT representation of the corresponding tokens. Recall that $i, j = 1, ..., m+n$. The final combined token representation, $\hat{z}^{p\to q}$, of q due to p, is calculated as a linear combination of the previous token representation and the corresponding current token attention weight. This combined token representation is used to calculate the representation v^p of node n^p as

$$v^q = (\sum_{q=1}^{l} \beta^{p\to q} \hat{z}^{p\to q})|z^q \tag{8}$$

where $|$ means the concatenate operator and $\beta^{p\to q}$ is calculated as

$$\beta^{p\to q} = softmax_p(MLP(z^q|\hat{z}^{p\to q})) \tag{9}$$

To summarize the procedure so far followed in the paper, the similarity between the previous representations of each neighboring nodes' token is used to calculate the attention weight for the edge-kernel of a particular neighboring node and it is used along with the concerned node's previous representation to develop the current representation of that node. Now on the basis of v^q, the label of the claim is determined using $P(y|n^q, G) = softmax_y(Linear(v^q))$. This way the label predicted for the claim is based on the information received from all the claim-evidence pairs.

Next step in the procedure is to calculate the "readout" module that combines the weighted label predicted on each node where weights are calculated according to how important a particular evidence is to the claim given the other claim-evidence pairs. In other words,

$$P(n^p|G) = softmax_p(Linear(\phi(n^p))) \tag{10}$$

where $\phi(n^p)$ is called the "readout" representation or the node-kernel representation of the node n^p and is calculated as,

$$\phi(n^p) = \frac{1}{m}\sum_{i=1}^{m} \vec{K}(M_i^{c\to e^p}) \tag{11}$$

where $(M_i^{c \to e^p}$ is calculated in a similar fashion as described for edge kernels with the difference that it is calculated between the representations of claim and the p^{th} evidence. This procedure translates to developing the node-kernel as a measure of similarity between the token representation of the claim and p^{th} evidence and then using this node-kernel to determine how important a particular evidence is for the given claim.

Finally, the label predicted according to each claim-evidence pair is weighted with the ”readout” module and the final labels are calculated according to equation(4). The model is trained using cross entropy loss between the predicted and the actual label.

3. Implementation and Evaluation

The focus of the implementation of this paper will be on (Clark et al., 2019). The code was made available at https://github.com/clarkkev/attention-analysis. I performed 6 experiments that can be broadly divided into two categories as described above. These will be discussed below.

The implementation starts with pre-training of BERT with 992 Wikipedia segments. The result you see below are developed from attention maps extracted from BERT during training implying the relations learnt by BERT were completely self-supervised.

Before being operated on by BERT, the segments needed to be put into the required format:

```
[CLS]1st paragraph[SEP]2nd paragraph[SEP]
```

Each segment provided to BERT was processed to have two consecutive paragraphs equivalent of 128 tokens. Now with this as input, attention map of each head in BERT was extracted. To be precise, attention map refers to, given a token in a segment provided to a particular attention head, which other token in that segment does that head attend to and with what intensity. Intensity here would refer to how large the attention weight is. The BERT used here comprises of 12 layers each with 12 attention heads. For the 1st experiment, the attention maps for 4 heads at 4 different layers for multiple tokens have been shown in Figure 1, 2 and 3 for three different segments.

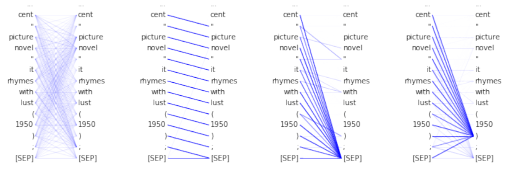

Figure 1. 256th segment

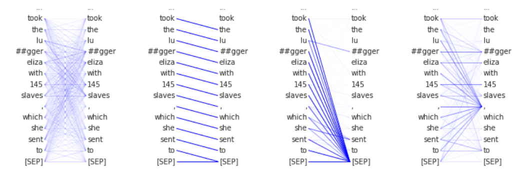

Figure 2. 196th segment

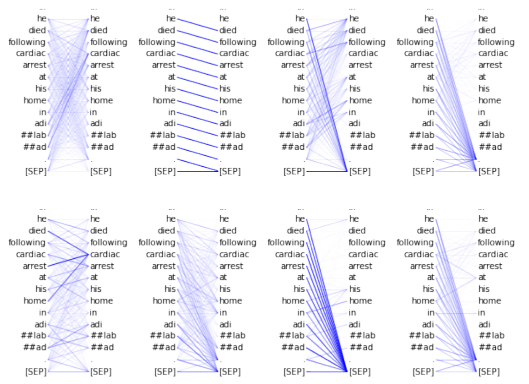

Figure 3. 325th segment

The heads belong to the layers, from left to right, 1, 3, 8 and 12 respectively so that the attention maps in the early layers, the middle layers and the later layers can be studied. In 3, the top picture is the attention map of the 1st head and the bottom picture is the attention map from the 12th head of the respective layers. The thing to notice here is how the model has learnt which word should be associated with which other word in a completely self-supervised manner.

The intensity of the link between the left and right columns of each head denote the weight of attention between those tokens. For the same segment, the first layer seems to focus equally on all the tokens provided. While by the third layer, the attention is accurately concentrated on the next token in the actual segment. By the 8th layer, almost all the attention, regardless of the segment, is focused on the [SEP] token and in the 12th layers, the weight of attention coefficient reduces but it is still the [SEP] token that is given most of the attention. This progression of learning as the tokens pass through the layers is noticeable.

Recall another observation from the paper that said heads are mostly attentive towards either the previous token or the next token. Through the 2nd experiment, it becomes evident. While for the 1st and the last layers, there are a few heads attending to the current token like ”1950” $\longrightarrow$ ”1950” in 1 1st layer and ”145” $\longrightarrow$ ”145” in 2, most of the significant attention weights are on the next and previous tokens in the segment. The last plot in 4 further strengthens this claim as it shows the average attention on next, previous and current token and the current token received the least amount of attention in all layers. This observation agrees with what happens in an actual language where each word is significant only in the context of other words in the sentence and helps develop a meaningful sentence.

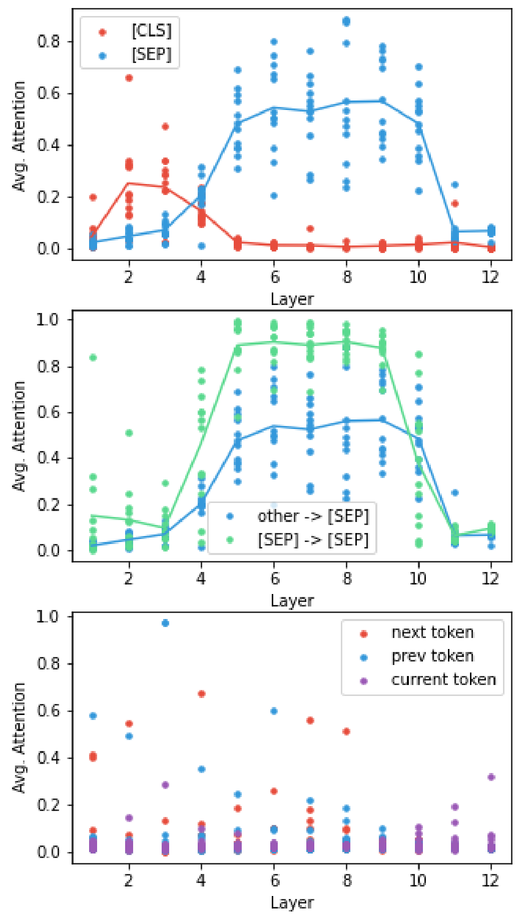

Figure 4. Average attention on various types of tokens

Another important observation with far reaching consequences found in these experiments was the distracted nature of these attention heads in general. The 3rd experiment confirms this as attentions on token of specific types are analysed. As shown in the first picture of Figure 5 the average attention entropy in the layers, specially in the central layers, is comparable to uniform attention. The plotted quantity is attention entropy, a high value of which indicates less concentrated attention. This consequence was also observed in (Liu et al., 2020) wherein they tried kernel attention to mitigate the drawback of dot product attention.

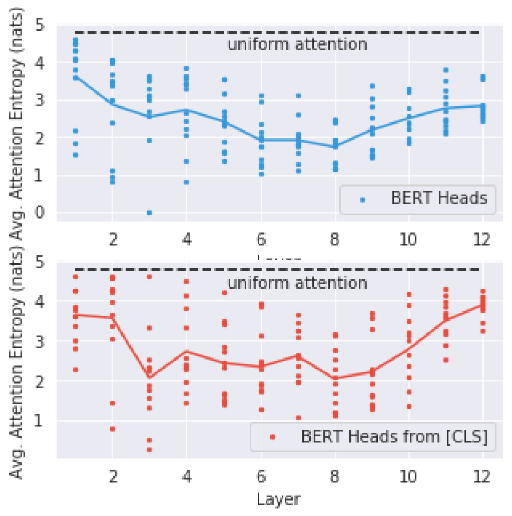

Figure 5. Average attention entropy

It is also worth noting that in the second picture of Figure 5, the last and the first layer has a very high attention entropy on the [CLS] token. As noted previously, this is significant as in at the beginning of learning process (in layer 1), it is not known as to which token should be attended to the most, while at the end of it (in layer 12), the vector representation thus produced should be a representation of all the tokens present.

In the 8th layer, highest attention weight led to the [SEP] token in all figures 1, 2 and 3. To investigate this behavior further, the average attention on each token is monitored. Specifically, the attention on special tokens [SEP] and [CLS] are studied.

In the top two pictures of Figure 4, it can be seen that the maximum attention in the middle layers was on the [SEP] tokens. Now recall that the input to BERT was 128 tokens comprised of two paragraphs with two [SEP] tokens. So statistically, the attention on [SEP] should be close to $\frac{1}{64}$. That clearly does not explain the behaviour in Figure 4.

The 4th experiment was, therefore, to further study this behaviour where the attention heads were specifically evaluated for dependency parsing on labelled data-sets, to investigate how semantic dependencies are learnt. This dependency is formally defined as given an input word in a sentence to a head, which other word from the sentence does the head assigns the largest weight to. This would imply that for a given input word, the head assigns the highest attention to that other word and thus drawing a dependency between those two words.

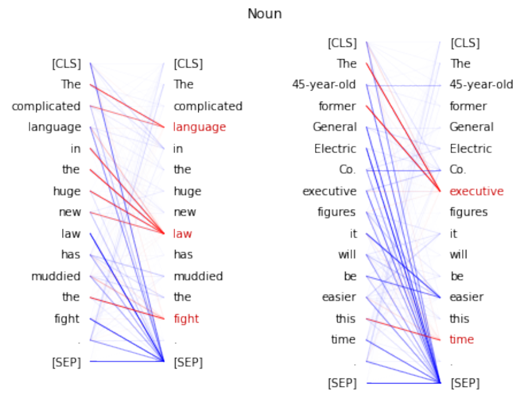

Figure 6. Curious case of attention on [SEP], Layer 8- Head 11

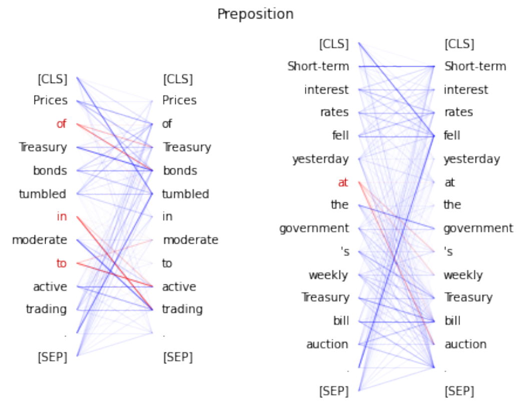

Figure 8. Attention for prepositions, Layer 9 - Head 6

As has been mentioned previously, for better explainability, word-word attention maps were developed instead of token-token. For example, the word *disappointed* is made up of two tokens, *dis* and *appointed*. To convert the token attention map to that of word for this example, the attention to *dis* and to *appointment* would be added as the attention to *disappointment* and for attention from *disappointment*, the mean of the attention from *dis* and that from *appointment* would be calculated. Proceeding this way, the dependencies captured by each head, for the segment provided to them, was mapped. As can be seen in Figure 6, the words on left that have nothing to attend to, attend to [SEP]. Therefore the provided hypothesis (Recall Section 2.2) on [SEP] being the default target of each head's attention stands true.

The 5th experiment consisted of probing the individual heads for attention maps and some more semantic dependencies learnt by the heads were discovered as shown in Figures 7 and 8. The [SEP] attention here is not displayed for clarity although their attention would be like the ones shown in Figure 1 - 3. It should be noted that such good learning of semantic dependencies were only found for some heads and that it is not a general pattern as can be seen in Figures 9, 10 and 11, where the learnt dependency is not so accurate.

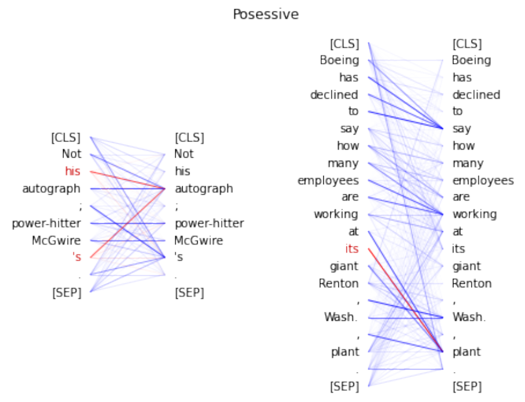

Figure 7. Attention for possessive nouns, Layer 7 - Head 6

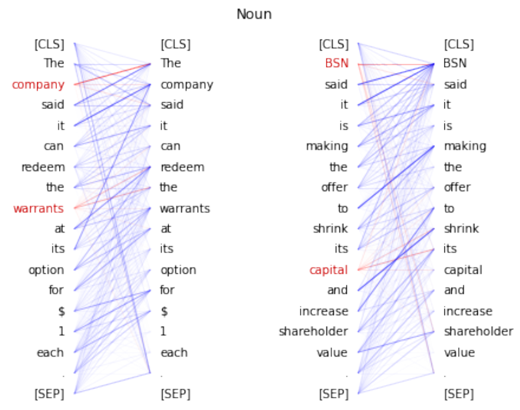

Figure 9. Incorrect attention for nouns, Layer 5 - Head 11

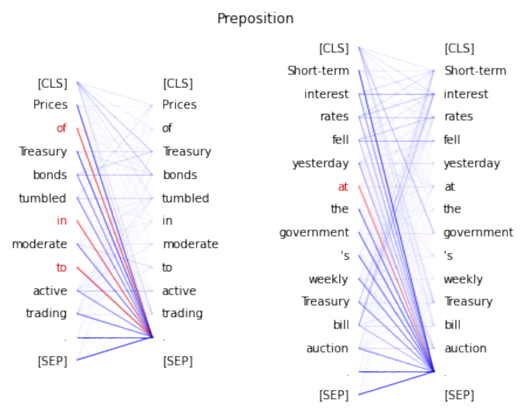

Figure 10. Incorrect attention for prepositions, Layer 3 - Head 6

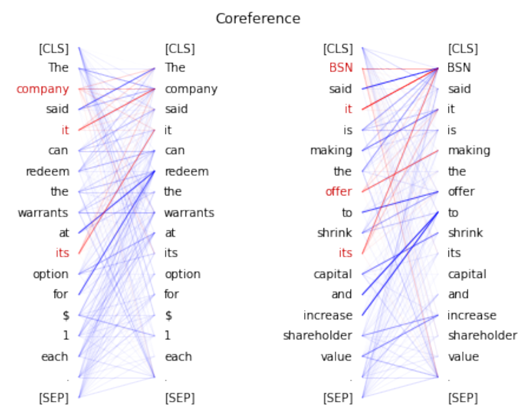

Figure 12. Coreferencing in BERT in Head 11 in Layer 7

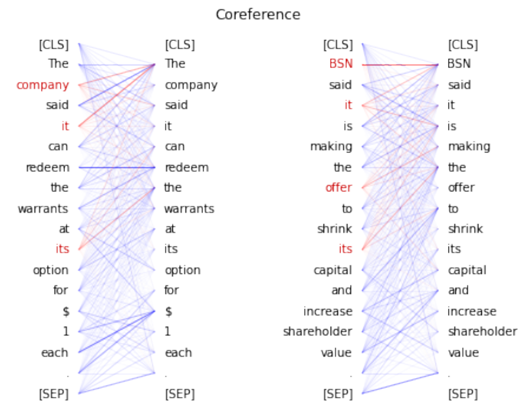

Figure 13. Incorrect coreferencing in Head 11 in Layer 2

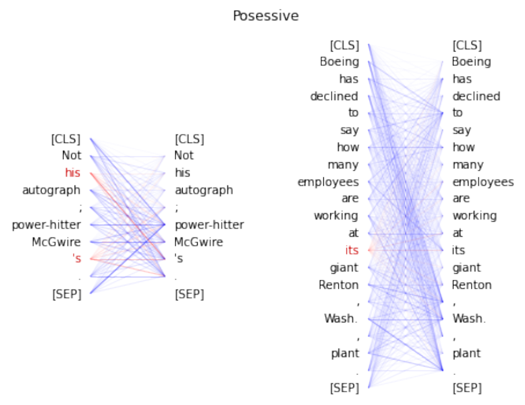

Figure 11. Incorrect attention for possessive nouns, Layer 12 - Head 6

The next sub-task was that of monitoring coreference resolution learnt in BERT. In language, when two different words refer to the same noun, one of them is called the antecedent (the one with the actual form) and the other is the proform. For example,in the sentence, *Henry was hungry so he ate*, *Henry* is the antecedent and *he* is the anaphor. The attention maps from BERT were tested for the same. As can be seen from Figure 13, BERT is more or less capable of capturing the coreference as in the attention graph, *company*, *it* and *its*, they all attend to company as they should. But again, this efficiency is not constant as can be seen from Figure **??**.

The last experiment was to verify if there are any similarities between the attention maps furnished by various heads. As described earlier, this was measured through the Jensen-Shannon divergence between the attention distributions and a corresponding two dimensional embedding was developed representing each attention head such that measure of the divergence is maintained. Accordingly, the heads found good with semantic abilities described above are plotted, that are layer 7-head 10 for nouns, layer 8-head 5 for preposition, layer 6-head 5 for possessive nouns and layer 4-head 3 for coreference. As can be seen in from the first plot in Figure 14, the above mentioned heads that learnt the semantics well are close together, so are the heads that attend broadly and the ones that attend to [SEP]. However, no pattern was found for the [CLS] token. This observation are on basis of hard thresholds as in for an entropy value of 3.8 and above, the heads were said to attend broadly, for an attention value of 0.6, the head was said to be focused on the [CLS] or the [SEP] token. In the second plot of 14, attention between the heads belonging to similar layers show up closer on the two-dimensional graph which mean their attention distributions must have been similar as well.

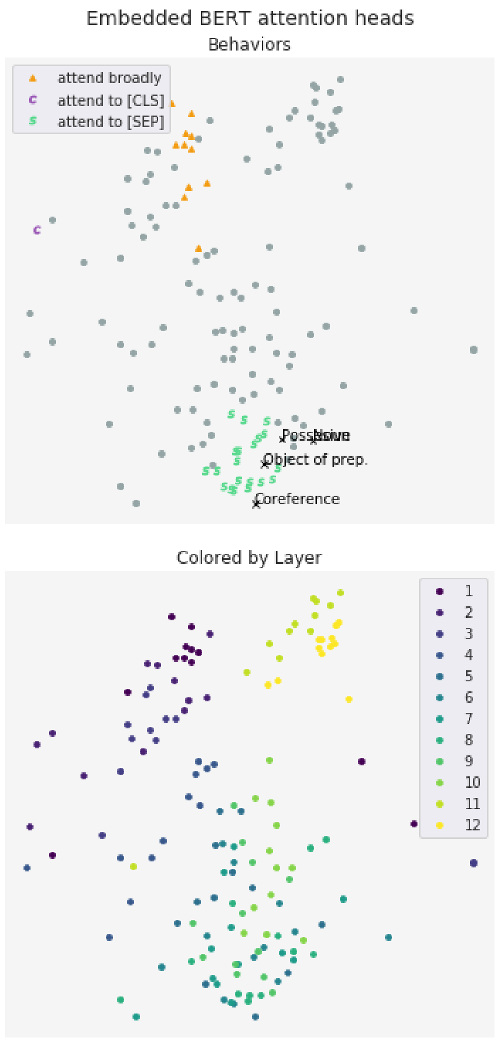

Figure 14. Clustering with respect to attention distribution

4. Discussion

Analysing the papers mentioned here, it is easy to see that (Liu et al., 2020) efficiently addresses the problems previously noted for attention and graph networks. This part of the section will analyse those.

As (Zhou et al., 2019) aims to solve the same problem as (Liu et al., 2020), we will first compare and contrast the methods used by each of them. Both the papers use the FEVER dataset to experiment with their models. While the document retrieval procedure is same in both, (Zhou et al., 2019) retrieves the sentences using relevance score with threshold and (Liu et al., 2020) implements BERT to project the hidden state representation of [CLS] into a ranking score with pairwise loss. The procedure of claim verification, as explained in both the subsections, use completely different techniques. While they both attempt to reason over all the evidences for a particular claim, the derivation of the attention coefficient in (Zhou et al., 2019) for the prediction of label per node is completely different than that in (Liu et al., 2020), as the latter uses edge-kernel attention for the same. Similarly for the evidence aggregator, (Zhou et al., 2019) uses simple aggregators such as attention, max and mean while (Liu et al., 2020) uses the node-kernel attention developed by (Knyazev et al., 2019) for the purpose. In its results, (Liu et al., 2020) quotes scores from (Zhou et al., 2019) for fact verification in using FEVER and LA dataset for claims that require both multiple and single evidences and reports the highest scores to be achieved by (Liu et al., 2020).

(Liu et al., 2020) further probes the function of the kernels used in KGAT. Recall that (Clark et al., 2019) reported that the attention heads in their model that were using dot product spent at most 10% of their attention mass on any single word, which translates to not having a focused attention the tokens that are more important that other. (Liu et al., 2020) also compares this focus for their kernel attentions, dot-product attention and uniform attention and found that kernel attention fares best in paying high attention to small number of tokens. Paying more attention to a selective number of features helps the model learn those better than the other features.

While (Liu et al., 2020) conducts a case study on a sample claim to analyse the attention distribution of the overall model on the claim and the evidence sentences thus retrieved, the attention distribution with respect to layer in KGAT, as analysed in (Jawahar et al., 2019), would be interesting to compare with that of BERT since both of them are after all used for developing language models.

The major improvement in KGAT over other models is the node kernel for calculating the weight of each evidence with respect to the claim while combining the label predictions from each node. This aspect of having attention on nodes has been well studied in (Knyazev et al., 2019). Much details have not been provided regarding how the methods specified in (Knyazev et al., 2019) were used in (Liu et al., 2020), but through the performance of the KGAT model it is highly likely that the directions given in the former study were followed.

References

Clark, K., Khandelwal, U., Levy, O., and Manning, C. D. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks* *for NLP*, pp. 276–286, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-4828. URL https://www.aclweb.org/anthology/W19-4828.pdf.

Jawahar, G., Sagot, B., and Seddah, D. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3651–3657, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1356. URL https://www.aclweb.org/anthology/P19-1356.pdf.

Knyazev, B., Taylor, G. W., and Amer, M. Understanding attention and generalization in graph neural networks. In Wallach, H., Larochelle, H., Beygelzimer, A., dAlché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 4202–4212. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/8673-understanding-attention-and-generalization-in-graph-neural-networks.pdf.

Liu, Z., Xiong, C., Sun, M., and Liu, Z. Fine-grained fact verification with kernel graph attention network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7342–7351, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.655. URL https://www.aclweb.org/anthology/2020.acl-main.655.pdf.

Zhou, J., Han, X., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. GEAR: Graph-based evidence aggregating and reasoning for fact verification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 892–901, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1085. URL https://www.aclweb.org/anthology/P19-1085.pdf.