Roles and Utilization of Attention Heads in Transformer-based Neural Language Models

Jae-young Jo
School of Computing, KAIST
Dingbro AI Research
kevin.jo@dingbro.ai

Sung-hyon Myaeng
School of Computing, KAIST
myaeng@kaist.ac.kr

Abstract

Sentence encoders based on the transformer architecture have shown promising results on various natural language tasks. The main impetus lies in the pre-trained neural language models that capture long-range dependencies among words, owing to multi-head attention that is unique in the architecture. However, little is known for how linguistic properties are processed, represented, and utilized for downstream tasks among hundreds of attention heads inside the pre-trained transformer-based model. For the initial goal of examining the roles of attention heads in handling a set of linguistic features, we conducted a set of experiments with ten probing tasks and three downstream tasks on four pre-trained transformer families (GPT, GPT2, BERT, and ELECTRA). Meaningful insights are shown through the lens of heat map visualization and utilized to propose a relatively simple sentence representation method that takes advantage of most influential attention heads, resulting in additional performance improvements on the downstream tasks.

1 Introduction

Sentence encoders in transformer architectures as in GPT, BERT (Vaswani et al., 2017; Radford, 2018; Devlin et al., 2019) and ELECTRA (Clark et al., 2020) have shown promising results on various natural language understanding (NLU) tasks, such as question answering, text entailment and natural language inference (NLI) (Bowman et al., 2015), owing to their pre-training capabilities in modeling languages.

The pre-training effects of the transformer-based approaches are known to be crucial for obtaining superior performance in various downstream NLU tasks. The main impetus lies in capturing long-range dependencies among words obtainable with bidirectional learning and self-attention (Devlin et al., 2019) and sufficiently varied corpora of a large quantity (Radford et al., 2019).

Despite all the recent successes of the transformer-based models, little is known for how linguistic properties are processed and represented internally when the architectures are used. Given that self-attention heads are unique in the family of transformer architectures, we attempt to answer the question of how basic linguistic properties are captured with the attention heads across the models and used for downstream tasks. Once we figure out the roles of attention heads in "storing" various linguistic properties, we should be able to modulate them to maximize the performance of the downstream tasks.

Given the motivation, we analyze several publicly available pre-trained transformer encoders (BERT, GPT, GPT2, and ELECTRA) trained with different model capacities ranging from 144 to 384 attention heads and 12 to 24 layers. Considering the output vector from each attention head of an encoder as a mini-sentence embedding, we examine whether certain linguistic properties are "stored" in embeddings among ten sentence probing tasks (Conneau and Kiela, 2018) that cover surface, syntactic, and semantic information and require different linguistic properties (*e.g.* the depth of a parsed sentence). Each of the probing tasks is treated as if it were a downstream task for the examination; a classifier is attached for each of the primitive linguistic properties. In order to predict the depth of the parse tree, for example, an n-ary classifier is connected, where n is the number of possible depths.

In order to aggregate and summarize the performance results out of all the attention heads, we construct an accuracy heat map for each probing task, where the patterns across layers and attention heads can be recognized easily. By examining the heat map, we can observe the patterns of how the attention heads contribute to the accuracy of each probing task, including whether an individual attention head is contributing to multiple linguistic features together or just specialized for a particular feature.

Aiming at producing improved sentence representation, we use the analysis result that allows for selecting and concatenating the outputs of superior attention heads. The sentence representations from the hidden layers and the top-n attention heads are compared to check whether using only influential attention heads selectively could help certain downstream tasks. This attempt is in contrast with the common approach of using the output of the last layers of a transformer-based encoder as the representation that is fed into a downstream task. Our hypothesis is those final representations from the top of the transformer-based encoders might not be the best not only in carrying primitive linguistic properties of the language but also for downstream tasks. All the source code is publicly available[1].

The major contribution of our research is twofold: 1) we suggest an analysis method which helps understand where linguistic properties are learned and represented along attention heads in transformer architectures and 2) we show that using analysis results, attention heads can be maximally utilized for performance gains during the fine-tuning process on the downstream tasks and for capturing linguistic properties.

2 Related Work

Several studies looked into the representations learned by a neural network for various language properties (Adi et al., 2016; Qian et al., 2016a,b). A similar line of work focused on learned linguistic features inside the word and sentence embeddings. They used downstream tasks in order to probe surface information, syntactic and semantic information (Shi et al., 2016; Conneau et al., 2018). Some recent work looked inside the sentence encoders with various depths, by analyzing the hidden states at a layer-level (Belinkov et al., 2017; Peters et al., 2018) and even at a neuron-level (Dalvi et al., 2018). Tenney et al. (2019a,b) attempted to understand linguistic characteristics learned in a series of pre-trained encoder models by jointly analyzing their behaviors across different NLP tasks.

For studying attention mechanisms, there have been two streams of work: 1) visual analysis of attention weights to associate various functionalities and 2) analysis of the characteristics of the output representations from individual attention heads.

For the first category, Vig and Jesse (2019) developed a visualization tool for attention weights of BERT and GPT2 and identified notable heads but without any quantitative analysis. Ghader and Monz (2017) showed the extent to which attention agrees with traditional alignments in neural machine translation (MT). Jain and Wallace (2019) and Brunner et al. (2019) on the other hand argued that attention rarely provides an explanation of model predictions. They showed through attention map analysis that attention weights frequently are not correlated with other measures of feature importance.

For the second category that attempts to discover various roles attention heads play, Raganato and Tiedemann (2018) studied the characteristics of individual attention heads from the transformer, pre-trained with an MT task and evaluated on a limited suite of linguistic tasks, POS tagging, NER tagging, and chunking. Similarly, Clark et al. (2019) showed that some attention heads are specialized for dependency parsing and coreference resolution. Michel et al. (2019) showed through an ablation study that some dedicated heads have a significant role in MT and revealed the dynamics of attention heads during the training process. Voita et al. (2019) provided a method to identify the major role of each attention head in a transformer model trained for MT. The two studies are limited to MT and a particular transformer model, BERT.

Unlike the recent studies mentioned above, our analysis is more comprehensive in its scope for generalizability. The analysis probes a variety of surface, syntactic, and semantic information at sentence levels with different transformer encoders pre-trained on language modeling tasks. More importantly, our work goes beyond an analysis and suggests a method of utilizing the analysis results for performance gains on several downstream tasks. It not only proposes a simple yet new method for the downstream tasks but also validates the analysis of the attention mechanisms. To the best of our knowledge, this is the first attempt to do an in-depth analysis of the seven recent pre-trained encoders for their internal workings in handling linguistic features, not to mention the newly proposed way for improvements on the downstream tasks.

[1]`https://github.com/heartcored98/transformer_anatomy`

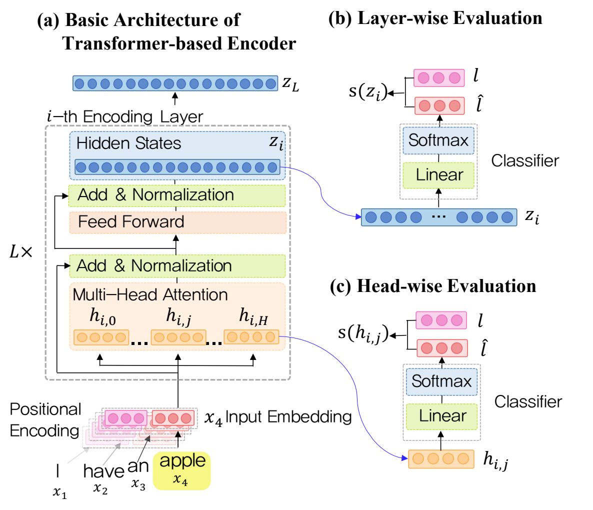

Figure 1: (a) Basic architecture of a transformer-based encoder. (b) Evaluation scheme for a hidden state z_i. (c) Evaluation scheme for an attention head output $h_{i,j}$. L and H denote the number of stacked encoding layers and the number of attention heads packed within each encoding layer, respectively.

3 Methodology

Consider a transformer-based encoder M, typically with a stack of L identical layers, each of which makes use of multi-head self-attention, and a two sub-layer feed-forward network coupled with layer normalization and residual connection (see Figure 1a). For a given input sequence $x = (x_1, x_2, \dots, x_n)$, each word embedding xis concatenated with a positional encoding and fed into the encoder layer to produce an attention head output $h_{i,j} \in R^{d_{head}}$ where i and j indicate the indices of the layer and the attention head, respectively. Then a series of sub-layers produce hidden states of the i-th encoding layer $z_i \in R^{d_{model}}$ for each encoder. For all pre-trained encoders, $d_{head} = 64$ and $d_{model} = H \times d_{head}$ where H is the number of attention heads per layer.

Since the transformer-based encoders encode the input sequence word by word, z_i and $h_{i,j}$ are produced individually for given word x_k along the input sequence x. In order to produce a sequence-level representation, we need to select one of the input representations of the sequence. Since the selection method depends on the chosen pre-trained model, we defer a detailed discussion to Section 4.1. For now, we assume z_i and $h_{i,j}$ have been already determined with the specific word chosen from the input sequence and consider it as the sentence-level representation.

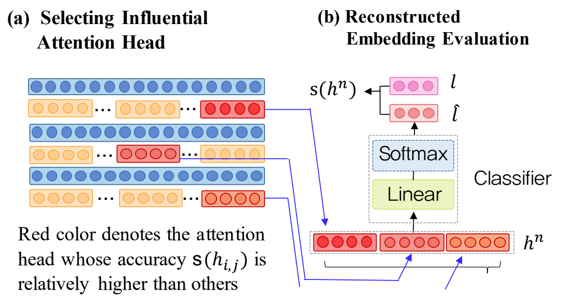
(a) Selecting Influential Attention Head
(b) Reconstructed Embedding Evaluation
Red color denotes the attention head whose accuracy $s(h_{i,j})$ is relatively higher than others
$s(h^n)$
l
$\hat{l}$
Softmax
Linear
Classifier
h^n

Figure 2: Selecting influential attention head output based on attention head-wise evaluation result. For example, assume colored three attention heads produce most superior representation then we concatenate the output from those attention head and use it as a sentence embedding.

3.1 Evaluating Hidden States on a Layer

Consider a classification task where the pre-trained encoder predicts a linguistic feature intended in a sentence probing task. Assume we have a labeled dataset containing pairs of a sentence and a linguistic property label (*e.g.* tense). For a given sentence x and a label l in the dataset, the pre-training model (*e.g.* BERT) encodes x and produces vectors corresponding to z_i and $h_{i,j}$.

Usually, only the vector from the last layer $z_{i=L}$ is used as the input feature representing the sentence for the classification task. However, in order to inspect the role of each internal layer for a linguistic property, we use $\{z_{i,l}, l\}$ for all i to train a logistic regression classifier on a train dataset and record classification accuracy $s(z_i)$ on a test dataset (see Figure 1b). Each accuracy score is then compared to the accuracy of the last layer, and then the best performance among the encoding layers is measured. We consider this comparison as a way of generating primitive evidence that hidden states from an internal layer provide more useful linguistic information than the representation from the last layer.

3.2 Evaluating Attention Heads

Similar to Section 3.1, we also train a logistic regression classifier on $\{h_{i,j}, l\}$ and record classification accuracy $s(h_{i,j})$ for all i and j. That is, every attention head is evaluated by feeding its own output vector to the classifier as a feature (see Figure 1c). We assume the more an attention head "stores" the information essential to the probing task, the higher its accuracy.

We construct a heat map of classification accu-

Encoder	L	H	$L \times H$	Parameters
GPT	12	12	144	110M
GPT2	12	12	144	117M
BERT$_{\text{BASE}}$	12	12	144	110M
BERT$_{\text{LARGE}}$	24	16	384	340M
ELECTRA$_{\text{SMALL}}$	12	4	48	14M
ELECTRA$_{\text{BASE}}$	12	12	144	110M
ELECTRA$_{\text{LARGE}}$	24	16	384	340M

Table 1: Specification of the seven pre-trained encoders: the numbers of encoding layers (L), attention heads per layer (H), all the attention heads used ($L \times H$) and trained parameters.

racy for attention heads on x-axis and layers on y-axis, so that we can easily identify the distribution of the excited attention heads for the linguistic property handled in the pre-trained model. The overall trend of a heat map indicates the extent to which the activation is widely distributed or localized across different layers and attention heads.

3.3 Using Influential Attention Heads

Given the analysis results, we now propose a method for generating a new sentence representation to improve not only the probing tasks but also other downstream tasks. New representations are tested within the chosen pre-trained models in this work but can be applied to all other transformer-based encoders.

Given an encoder model M, we sort the attention heads along with their classification 'validation' accuracy $s(h_{i,j})$ measured on a validation dataset (in order to prevent look-ahead bias during the selection process) for a given task, based on the attention head-wise evaluation method as in Section 3.2. Then top-n attention heads are selected and simply concatenated (see Figure 2) to form a new representation. We expect that the resulting vector $h^n \in R^{n \times d_{head}}$ would be able to store more precious information for the task than the vectors constructed out of other attention heads since it consists of superior attention heads.

In order to make comparisons against the embeddings from different encoding layers, we also train the classifier with $\{h^n, l\}$ and record the corresponding classification 'test' accuracy $s(h^n)$ measured on the test dataset. For fair comparisons, however, we set n to H (the number of attention heads per layer) so that reconstructed sentence embedding h^n could have the same dimension to that of hidden states, d_{model}.

Tasks	# Classes	Task Description
Length	6	Predict input sequence length
WordContent	1000	Find words in a sentence
Depth	8	Predict maximum depth of syntactic tree
TopConst	20	Predict top-constituents
BigramShift	2	Detect bigram order perturbation
Tense	2	Predict main verb's tense
SubjNum	2	Predict whether a subj is plural
ObjNum	2	Predict whether an obj is plural
OddManOut	2	Detect noun or verb perturbation
CoordInversion	2	Detect clausal order perturbation

Table 2: Summary of sentence probing tasks. Each task consists of 100k train and 10k test samples.

4 Attention Head-wise Analysis

4.1 Pre-trained Transformer Encoders

We ran experiments for seven different encoders with unique characteristics, as shown in Table 1. GPT (Radford, 2018) was trained by basic Language Modeling (LM) on the BookCorpus dataset. GPT2 (Radford et al., 2019) was originally trained with the largest model capacity (1.5B parameters) with massive text dataset and LM, but we select base model for fair comparison. BERT (Devlin et al., 2019), which adopted masked LM (MLM) with next sentence prediction (NSP) for better contextualized embedding, was trained on Book-Corpus and English Wikipedia datasets. The most recent one, ELECTRA, was trained with replaced token detection (RTD) in the generator-discriminator mode.

For GPT and GPT2, we pulled the representative sentence embedding z_i and $h_{i,j}$ from the last input token with Byte-Pair Encoding tokenizer (Sennrich et al., 2016). For the BERT and ELECTRA family, we appended a special token <CLS>, which was originally designed to train sentence representations, in front of every input sequence and pulled the sentence embedding from it, using WordPiece tokenizer (Wu et al., 2016). Also, the implementation of the all transformers in our work are utilized from the Huggingface's transformers library (Wolf et al., 2019).

4.2 Evaluation on Sentence Probing Tasks

Ten sentence probing tasks enable us to check whether the sentence embeddings generated by the encoders store the linguistic properties specific to the individual tasks. Table 2 shows a description of each probing task with its number of classes, roughly indicating the difficulty of the task. For each probing task, we evaluated performance of three types of representation; $s(z_i)$, $s(h_{i,j})$ and

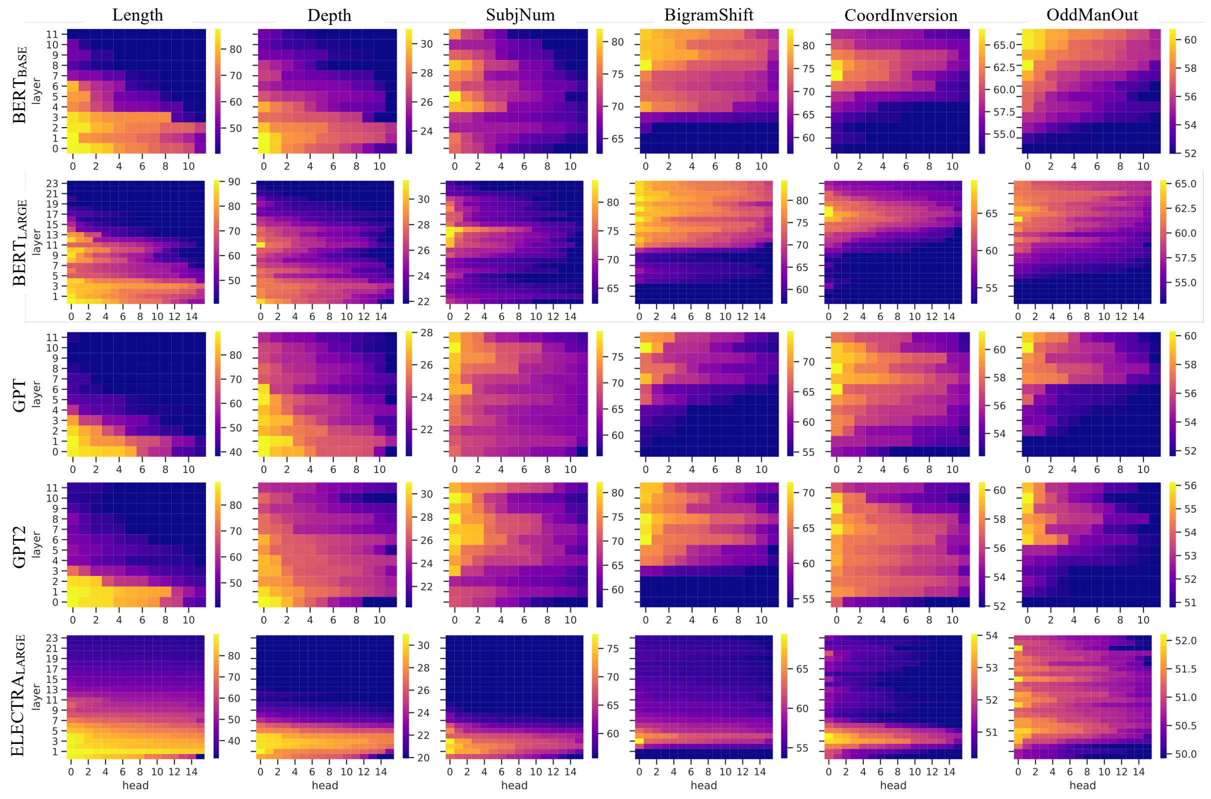

Figure 3: Heat maps of attention head-wise evaluation on sentence probing tasks. The rows correspond to the five pre-trained encoders (BERT$_{\mathrm{BASE}}$, BERT$_{\mathrm{LARGE}}$, GPT, GPT2, and ELECTRA$_{\mathrm{LARGE}}$ from the top). The six columns correspond to six tasks (Length, Depth, SubjNum, BigramShift, CoordInversion, and OddManOut, from the left). In each heat map, x-axis and y-axis show the index values of the attention heads and the layer numbers (the lower, the closer to the initial input), respectively. The brighter the color, the higher the accuracy for the attention head and hence more critical for the task. Note that the attention heads in the same layer are ordered by their classification accuracy values (i.e., an attention head with the highest accuracy on a layer is at the left-most location).

$s(h^n)$ for a given pre-trained encoder by training the simple classifier with 256 batch size on the RMSProp optimizer (details on Appendix A).

4.3 Heat maps for Roles of Attention Heads

After measuring the classification accuracy for using the representation from each attention head, $s(h_{i,j})$ for all i,j, we created a heat map showing the accuracy distribution for a pre-trained encoder and a sentence probing task. Figure 3 shows 30 heat maps arranged for seven pre-trained encoders and six sentence probing tasks (full results are shown in Appendix B). For each heat map, the brighter the color in a region, the higher the accuracy is for the corresponding attention heads.

Comparing the heat maps along with the different probing tasks for an encoder, we can see that the influential attention heads with bright colors appear in different layers, either localized or distributed. This indicates that the information related to different tasks is processed at different locations and with different levels of association among attention heads. For the Length and Depth tasks, requiring surface and syntactic information, for example, the accuracy of the heads in the lower layers starts to diminish from the mid-upper layers.

On the other hand, the attention heads in the mid-layers are activated for SubjNum and CoordInversion, which are more or less syntactic information. For BigramShift and OddManOut, which are more semantic, the attention heads along the upper layers are mostly activated. These results provide more detailed analyses and meaningful insights regarding the behavior of attention heads on different layers than the empirical results of Raganato and Tiedemann (2018) who shows the attention heads in lower and upper layers of the basic transformer tend to embed syntactic information and semantic information, respectively. More interestingly, the BigramShift and OddManOut heat maps show that all of the five encoder models represent word orders and verb/noun contexts starting from the

Tasks	Encoder					
	$\text{BERT}_{\text{BASE}}$			$\text{BERT}_{\text{LARGE}}$		
	last layer	best layer	top-12 heads	last layer	best layer	top-16 heads
Length	58.0	87.8	**95.0**	54.8	94.4	**95.2**
WordContent	25.2	25.2	**73.1**	12.2	32.2	**79.8**
Depth	29.8	31.7	**38.3**	27.8	33.3	**39.5**
TopConst	69.8	74.7	**84.2**	62.8	78.2	**85.6**
BShift	78.1	78.1	**88.3**	77.2	81.1	**90.9**
Tense	86.0	87.0	**89.0**	85.6	86.7	**88.9**
SubjNum	82.0	84.7	**88.2**	80.0	87.9	**90.5**
ObjNum	75.4	75.4	**83.4**	64.2	78.0	**84.4**
OddManOut	59.6	59.6	**65.1**	55.5	59.2	**69.0**
CoordInv	65.5	65.9	**74.6**	64.9	70.9	**78.5**

Table 3: A summary of the probing tasks for three different embedding methods used in the pre-trained BERT architectures.

mid-layers.

Comparing the heat maps along with the transformer types, we can observe that the heatmaps within the same family show similar patterns, while those from different families tend to show different distributions of the superior attention heads. For example, the GPT family tends to show cooperation with a larger number of attention heads for the SubjNum and CoordInversion tasks while the BERT family consists of only a few "well-educated" attention heads. In the case of BigramShift and OddManOut, the majority of upper attention heads of the BERT family are more strongly associated with word order and verb/noun meanings with higher accuracy than those of the GPT family.

Interestingly, $\text{ELECTRA}_{\text{LARGE}}$ shows unique patterns for most of the probing tasks; high-performance heads are located on lower layers except for OddManOut, whereas the heads on the lower layers do not seem to deal with information for the probing tasks. $\text{ELECTRA}_{\text{SMALL}}$ and $\text{ELECTRA}_{\text{BASE}}$ model have similar heat maps (see Appendix B), but the $\text{ELECTRA}_{\text{LARGE}}$ model is totally different from them. These tendency implies that the learning behaviors on the attention heads are not strictly similar among each other for the same pre-training tasks even with the same architecture.

4.4 Selecting Influential Attention Heads

Having observed that different attention heads on different layers play their roles for different probing tasks, we devised a method of producing new embeddings as in 3.3 and ran an experiment to compare it against two baselines for the ten probing tasks. Table 3 reports on a comparison result of three embeddings constructed by the BERT family: the last layer $z_{i=L}$, the best-performing layer z_{best}, and the reconstructed sentence embedding $h^{n=H}$ for each task and each pre-trained encoder (full results are in Appendix B). Comparing the accuracy between the last and best layers, we observe that the last layer is no better than the "best" layers for any of the probing tasks. From this, we can infer that certain linguistic features are dominantly processed on earlier layers and no further on later layers.

The performance comparison between using the output of the "best" layer and the reconstructed sentence embedding (proposed) clearly shows that classification accuracy is increased significantly (19.22% in median) with the proposed method for almost all the tasks. It strongly supports that the proposed method can be employed to discover superior attention heads that can make up the final representation for processing specific linguistic information. Note that the newly constructed sentence embeddings consist of attention head outputs only. Our results imply that these embeddings might possess substantial information as much as the hidden states of the layers, which are produced by passing through the multi-head attention layers and the feed-forward network.

5 Boosting Downstream Task

5.1 Downstream Tasks from GLUE

We evaluated the new embedding construction method for more complex tasks in order to see whether it extracts not only simple linguistic features but also rich sentence features from the pre-trained encoder for such tasks. Three downstream tasks (MRPC, STS-B, and SST-2) were selected from the General Language Understanding Evaluation (GLUE) benchmark, which has the most widely used datasets for evaluating language-oriented task performances.

MRPC Microsoft Research Paraphrase Corpus consists of 4.1k train and 1.7k test sentence pairs automatically extracted from online news sources, with human annotations for whether the sentences in the pair are semantically equivalent (Dolan and Brockett, 2005).

STS-B The Semantic Textual Similarity Benchmark is a collection of 5.7k train and 1.4k test sentence pairs drawn from news headlines and other sources (Cer et al., 2017). They were annotated with a score from 1 to 5 denoting how similar the

Tasks	BERT$_{BASE}$ last layer	BERT$_{BASE}$ best layer	BERT$_{BASE}$ top-12 heads	BERT$_{LARGE}$ last layer	BERT$_{LARGE}$ best layer	BERT$_{LARGE}$ top-16 heads
MRPC (F1)	88.0	88.2	**88.9**	89.3	88.6	**91.4**
MRPC (Acc)	82.4	83.1	**84.6**	84.6	84.1	**87.7**
STS-B (P)*	88.2	74.6	**88.6**	**89.5**	54.8	89.4
STS-B (S)	87.9	73.5	**88.3**	**89.1**	53.6	88.7
SST-2 (Acc)	92.9	92.4	**93.1**	94.0	92.9	**94.5**

Table 4: A summary of three downstream tasks on dev set for the ordinary fine-tuning method using the last layer, best layer, and the proposed method of using top-n attention heads. The reported scores are the median over 5 random restarts. (* P and S denote the pearson score and spearman score, respectively.)

two sentences are for their semantics.

SST-2 The Stanford Sentiment Treebank is a binary single-sentence classification task consisting of sentences extracted from movie reviews with human annotations of their sentiment (Socher et al., 2013) with 67k train and 1.8k test samples. It was designed to predict the sentiment score for a given sentence in binary scales.

5.2 Fine-tuning with Influential Heads

First, we evaluated each of the attention heads on the three downstream tasks, following the procedure in Section 4.2. Using the head-wise evaluation results, we again reconstructed sentence embeddings from the output vectors of superior attention heads and use them as input representations for the downstream task classifier. Since pre-trained transformer encoders are usually fine-tuned when applied to the downstream tasks, we unfroze the parameters of the pre-trained encoder and fine-tuned both the classifier and the encoder, end to end. Also, we conducted regular fine-tuning experiments by adding a classifier on the top of the last hidden vectors for each pre-trained encoder. We use a batch size of 32 with a learning rate of 2e-5 and fine-tune for 3 epochs over the data for all the three downstream tasks, following the fine-tuning procedure in (Devlin et al., 2019). Each experiment is repeated five times with different random seeds to provide fair comparisons against the performance variance of fine-tuning on small datasets.

The results are presented in Table 4. Both BERT$_{BASE}$ and BERT$_{LARGE}$ obtained additional performance gains, 0.82% and 1.04% points for the base and large models, respectively, over the model with the ordinary last-layer fine-tuning. We find that BERT$_{LARGE}$ receives an additional performance gain on the MRPC task by 2.1% and 3.1% point improvements on F1 and accuracy, respectively. Fine-tuning with attention heads only gives a slightly negative result on STS-B with BERT$_{LARGE}$. Fine-tuning with the best-layer did not provide consistent performance increment. It is noteworthy that the performance of an already pre-trained encoder model could be further improved by simply pulling the output vectors from the influential attention heads.

6 Discussion

6.1 Heat Map Variations along Fine-tuning

In order to investigate the impact of the fine-tuning process toward the internal attention heads, we also conducted the attention head-wise evaluation on each encoder after three epochs of the fine-tuning process. Our question was whether the influential attention heads at the initial pre-trained state would remain superior after the fine-tuning or the spot of influential heads would be shifted toward the last layer.

The results are presented in Figure 4. First, we again observe that the regions of the influential heads vary among the downstream tasks. In the MRPC task, influential heads are distributed across the entire layers and heads, but the ones with the SST-2 task are highly concentrated toward the very upper layer. Notably, the heat maps of the STS-B task are unusual in that there are two influential regions in the lower (first 25˜30% layers) and the upper layers. We can also observe that the overall heat map patterns are stretched while the model capacity is increased, as reported in (Tenney et al., 2019a). From the way feature vectors are pulled from the encoder, we observe that fine-tuning with the reconstructed sentence embeddings obtained from the top-n attention heads results in the smoother heatmap amplification, especially with the BERT$_{LARGE}$ model.

The most interesting result is that the intensity (performance) of the initial heatmaps are amplified after experiencing the fine-tuning process while preserving overall distribution patterns. Another phenomenon is that the attention heads adjacent to the superior ones also give a slight performance increase. These results imply that the fine-tuning process leverages the initial superior attention heads regardless of their corresponding locations inside the model rather than trains arbitrary attention heads. This behavior might be the reason for explaining

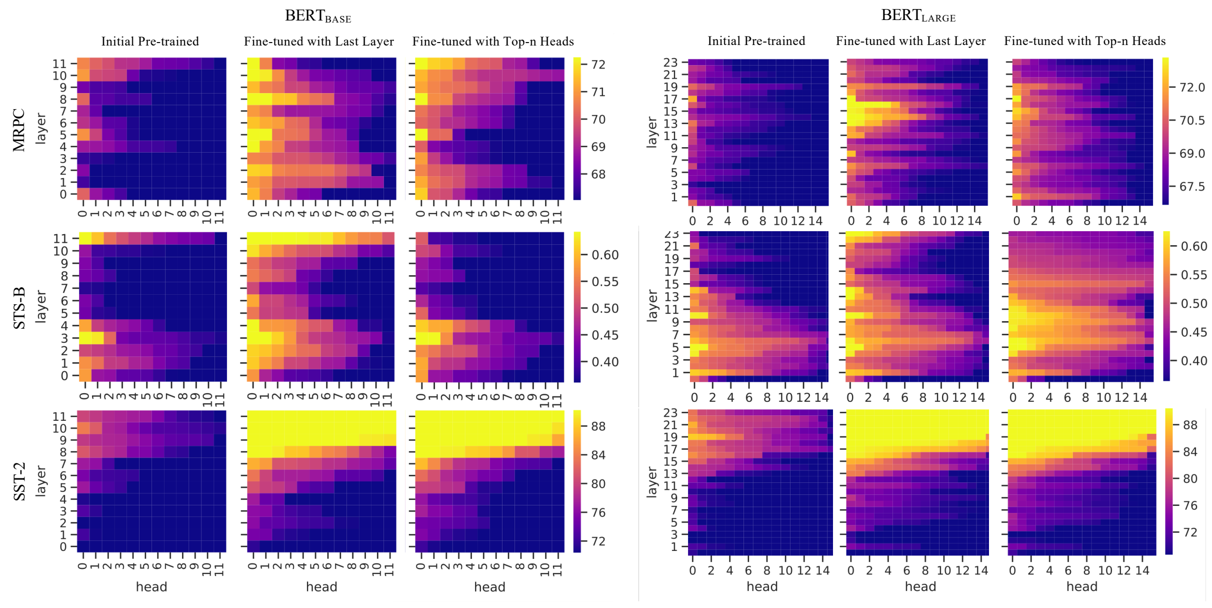

Figure 4: Heat maps of attention head-wise evaluation on downstream tasks. The rows correspond to the three tasks (MRPC, STS-B and SST-2 from the top). The first three column groups are evaluation results of BERT$_{\text{BASE}}$ and second three column groups are evaluation results of BERT$_{\text{LARGE}}$. Each column groups correspond to the initial pre-trained state, fine-tuned with last layer and fine-tuned with top-n attention heads, respectively. In each heat map is drawn following the procedure of Figure 3. Note that the heat maps in the same row within the same encoder model share the same color bar range in order to compare performance changes.

the additional performance increment on the downstream tasks. We conjecture that our reconstruction method could act as a partial residual connection as in DenseNet (Huang et al., 2017) during the fine-tuning process by feeding the reconstructed embedding to the input of the classifier which creates the direct gradient flow from the final objective loss of downstream tasks toward the internal superior attention heads. We believe that further work by varying the number of concatenated the attention heads (especially, $n > H$) would provide additional performance gain.

6.2 Syntactic-Semantic Exclusivity

Our analysis so far concentrated on the distribution of the influential attention heads on different layers for given task as a way of differentiating their roles for individual tasks. A pattern we observed was that different number of heads are influential and that upper, lower, or all the layers tend to be influential, depending on the linguistic tasks. Our next question is whether individual heads on different layers are ”responsible” for processing syntactic or semantic properties exclusively or in a coordinating fashion. In order to observe the performance of attention head $h_{i,j}$ for syntactic and semantic tasks, we define a score for handling syntactic capabilities as an average of test accuracy scores, $s(h_{i,j})$, from the [*Depth, TopConstituents, BigramShift*] group and that for semantic capabilities from the [*Tense, SubjNumber, ObjNumber, OddManOut, CoordinationInversion*] group. We omit the accuracy results from the surface information group since they it is difficult to lablem as syntactic or semantic.

Figure 5 shows the syntactic-semantic score distributions of the attention heads for different pre-trained transformer models. Each attention head seems to handle both syntactic and semantic information in a balanced way. This is interesting because different attention heads or layers are often more influential for many linguistic tasks. When averaged together over the tasks for either the syntactic or semantic group, however, it appears that processing syntactic and semantic information is shared by individual heads and layers. There is a tendency that the lower the layer, the less influential on syntactic and semantic processing. However, this tendency is not observed in the large models. For BERT$_{\text{LARGE}}$, the highest layers (purple colors) contribute less for both syntactic and semantic properties. For ELECTRA$_{\text{LARGE}}$, the purple heads contribute the least. It re-confirms our hypothesis that using the last layer representation is not always the best. The linear relationship between syntac-

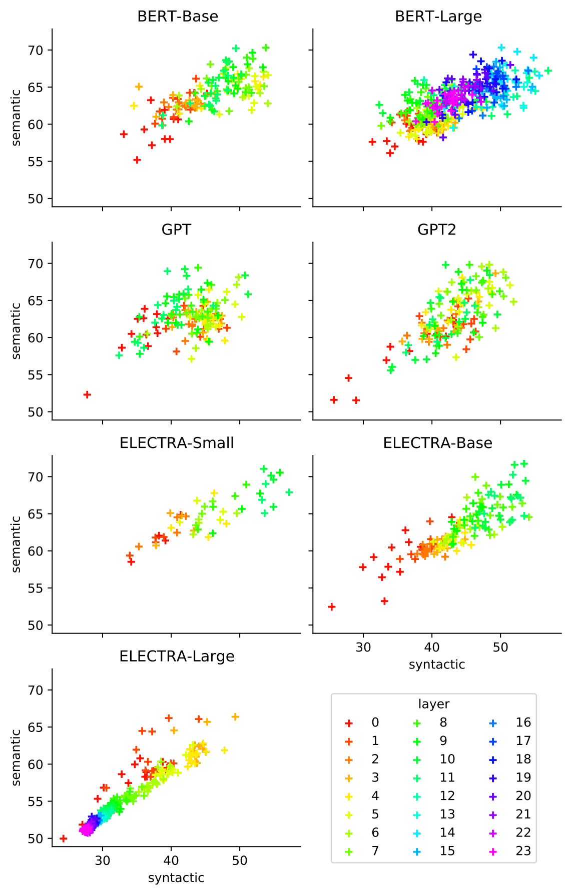

Figure 5: A distribution of syntactic and semantic scores of the attention heads. In each scatter plot, x-axis and y-axis show the syntactic and semantic scores, respectively. The hue of a point represents the layer to which the corresponding attention head belongs.

tic and semantic processing capabilities across the heads is considered a new finding. Although different layers and heads tend to play stronger or weaker roles for different linguistic properties as shown in the heat maps, they contribute to both syntactic and semantic processing in a well balanced way.

7 Conclusion

While recent research demonstrated the capability of the transformer-based encoders for generating rich sentence representations, the roles of individual self-attention heads were hardly unknown. Furthermore, little is known for whether and how we can utilize them for better capturing linguistic properties and eventually improving the performance of downstream tasks for which the embeddings are constructed.

One of the major contributions of this paper is to fill the void by inspecting where and how the attention heads are "trained" internally for classification tasks corresponding to different linguistic properties and for the downstream tasks. The analysis results clearly show a tendency through the comprehensive heat maps that syntactic and semantic information is mainly handled from the lower layers to the upper layers. We also showed that understanding the roles of attention heads in handling task-specific information can help to develop adaptive sentence representations, by selecting influential attention heads and testing them for the three downstream tasks. The additional performance gains obtained by the simple method show that this approach of using the anatomy of the transformer models and the attention heads is promising in utilizing expensive pre-trained transformer models to their maximal extent.

Furthermore, we explored how the hundreds of attention heads underwent performance variation during the fine-tuning process on the downstream tasks, revealing the internal behaviors with the proposed analysis method. The analysis of syntactic-semantic score distributions revealed that individual attention heads capture both syntactic and semantic information. It also showed that the amount of both syntactic and semantic information handled by the heads vary from layer to layer, sometimes showing that the last layer contributes much less especially with large models.

While the empirical results are strong, additional work remains to further our understanding of the internal workings of the transformer architecture and its role in building such strong language models for a variety of tasks. Immediate attention should be paid to the investigation of how heat maps would vary during the extensive pre-training so that we have a better understanding of the dynamics of the learning processes.

Acknowledgments

This work was supported by Institute for Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2013-0-00131, Development of Knowledge Evolutionary WiseQA Platform Technology for Human Knowledge Augmented Services). We are grateful for the support of the GPU server to IdeaFactory, Startup KAIST. We also appreciate Kyubyong Park, Seongok Ryu, and YJ for reviewing the earlier version of this paper.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *CoRR*, abs/1608.04207.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872, Vancouver, Canada. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Gino Brunner, Yang Liu, Damián Pascual, Oliver Richter, and Roger Wattenhofer. 2019. On the validity of self-attention as explanation in transformer models. *arXiv preprint arXiv:1908.04211*.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of bert's attention. *CoRR*, abs/1906.04341.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia. Association for Computational Linguistics.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, Yonatan Belinkov, Anthony Bau, and James R. Glass. 2018. What is one grain of sand in the desert? analyzing individual neurons in deep nlp models. In *AAAI*.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Hamidreza Ghader and Christof Monz. 2017. What does attention in neural machine translation pay attention to? In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 30–39, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. Densely connected convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. *CoRR*, abs/1902.10186.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 14014–14024. Curran Associates, Inc.

Matthew Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018. Dissecting contextual word embeddings: Architecture and representation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Brussels, Belgium. Association for Computational Linguistics.

Peng Qian, Xipeng Qiu, and Xuanjing Huang. 2016a. Analyzing linguistic knowledge in sequential model of sentence. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 826–835, Austin, Texas. Association for Computational Linguistics.

Peng Qian, Xipeng Qiu, and Xuanjing Huang. 2016b. Investigating language universal and specific properties in word embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1478–1488, Berlin, Germany. Association for Computational Linguistics.

Alec Radford. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Alessandro Raganato and Jörg Tiedemann. 2018. An analysis of encoder representations in transformer-based machine translation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 287–297, Brussels, Belgium. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Xing Shi, Inkit Padhi, and Kevin Knight. 2016. Does string-based neural MT learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1526–1534, Austin, Texas. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019a. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019b. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

A Training and Evaluation Details

A.1 Pre-trained Transformer

Throughout the entire experiments, we mainly used huggingface's seven pre-trained transformers[2], implemented with Pytorch. However, since the original implemented models do not return the output vectors of the internal attention heads, we developed the wrapper class that enables extracting the output vectors from the created pre-trained model objects. The implementation details and procedures for replicating the experimental results are described in our repository[3]

A.2 Probing Task Benchmark

We utilized the SentEval toolkit[4] for both probing and downstream tasks. The probing task results reported in the main text are obtained with a logistic regression layer attached to the pooled output vector from the transformer. We trained the classifier with the batch size of 256 for all the experiments, *freezing* the parameters of the transformer. A RMSProp optimizer is used with the learning rate of 0.1. Only the L2 regularization is tuned among [10-5, 10-4, 10-3, 10-2, 10-1]. During training, we monitor the validation accuracy and stop the training process when it does not improve for the previous 3 epochs (tenacity=3). Also, each classifier is tested with 5-fold cross validation. In section 3.3, validation accuracy $s(h_{i,j})$ is measured from the five-validation sets partitioned in a mutually exclusive way and averaged.

[2]https://github.com/huggingface/transformers

[3]https://github.com/heartcored98/transformer_anatomy.

[4]https://github.com/facebookresearch/SentEval

A.3 Downstream Task Benchmark

The downstream task results reported in the main text are obtained with a logistic regression layer attached to the pooled output vector from the transformer. We trained the classifier with the batch size of 256 for all experiments, *freezing* the parameter of transformer, following the same procedure of A.2. Note that the metric for the STS-B task is Pearson and Spearman scores. Therefore we measured the validation Pearson score instead of validation accuracy for choosing influential attention heads for the STS-B task.

A.4 Fine-tuning on Downstream Tasks

During the fine-tuning process with one of the three different pooling methods (last-layer, best-layer, and top-n heads), we attached an additional linear layer with a dropout layer (dropout rate=0.1) and Tanh activation function, following the pooler architecture implemented in Huggingface's transformer. Then the logistic regression layer is attached to the activation function. We trained the classifier with the batch size of 32 with a learning rate of 2e-5 with three epochs for all the experiments, *unfreezing* all the parameters of the transformer and the regressor. Each experiment is repeated five times with different random seeds to provide fair comparisons against the performance variance of the fine-tuning process conducted on small datasets.

B Head-Wise Evaluation Results with Probing Tasks

The performance variation of the probing tasks is shown in Table 5 that provides full experimental results with BERT$_{\text{BASE}}$, BERT$_{\text{LARGE}}$, GPT and GPT2.

C Head-wise Evaluation Heatmaps

Since Figure 3 provides partial results only, we provide Figure 6 and 7 here to show the full experimental results with and without sorted attention heads on the same layer. The former helps understanding how the influential heads are gathered for their strengths while the latter is useful for understanding how various linguistic capabilities are supported in association by a particular attention head.

Task Group	Task	Encoder											
		BERT$_{\text{BASE}}$			BERT$_{\text{LARGE}}$			GPT			GPT2		
		last layer	best layer	top-12 heads	last layer	best layer	top-16 heads	last layer	best layer	top-12 heads	last layer	best layer	top-12 heads
Surface	Length	58.0	87.8 (51.5)	**95.0 (64.0)**	54.8	94.4 (72.2)	**95.2** (73.6)	52.2	**96.4 (84.7)**	96.2 (84.3)	57.8	88.9 (53.9)	**92.8 (60.6)**
	WordContent	25.2	25.2 (0.0)	**73.1 (190.3)**	12.2	32.2 (165.2)	**79.8** (556.4)	35.3	35.3 (0.0)	**71.3 (102.0)**	37.5	37.5 (0.0)	**71.0 (89.5)**
Syntactic	Depth	29.8	31.7 (6.5)	**38.3 (28.7)**	27.8	33.3 (19.6)	**39.5** (41.9)	27.2	30.6 (12.2)	**38.9 (42.8)**	28.0	31.0 (10.6)	**40.0 (42.7)**
	TopConstituents	69.8	74.7 (7.0)	**84.2 (20.6)**	62.8	78.2 (24.6)	**85.6** (36.4)	53.0	65.1 (22.8)	**79.5 (50.0)**	57.3	63.1 (10.2)	**82.8 (44.5)**
	BigramShift	78.1	78.1 (0.0)	**88.3 (13.1)**	77.2	81.1 (5.1)	**90.9** (17.8)	69.3	69.3 (0.0)	**80.7 (16.5)**	68.8	70.5 (2.5)	**78.8 (14.6)**
Semantic	Tense	86.0	87.0 (1.2)	**89.0 (3.6)**	85.6	86.7 (1.2)	**88.9** (3.8)	88.6	88.6 (0.0)	**89.0 (0.5)**	88.2	88.6 (0.5)	**89.3 (1.2)**
	SubjNumber	82.0	84.7 (3.4)	**88.2 (7.6)**	80.0	87.9 (9.9)	**90.5** (13.0)	78.8	78.8 (0.0)	**84.2 (6.8)**	83.5	83.5 (0.0)	**87.7 (5.1)**
	ObjNumber	75.4	75.4 (0.0)	**83.4 (10.6)**	64.2	78.0 (21.5)	**84.4** (31.4)	71.2	71.9 (0.9)	**80.5 (13.0)**	70.2	74.1 (5.5)	**82.5 (17.5)**
	OddManOut	59.6	59.6 (0.0)	**65.1 (9.3)**	55.5	59.2 (6.6)	**69.0** (24.2)	**55.0**	58.2 (5.9)	**63.0 (14.6)**	54.6	54.9 (0.6)	**5.5 (1.6)**
	CoordInversion	65.5	65.9 (0.5)	**74.6 (13.8)**	64.9	70.9 (9.2)	**78.5** (20.9)	57.7	60.7 (5.1)	**70.2 (21.6)**	58.3	60.0 (2.9)	**67.6 (15.9)**

Table 5: A summary of the probing tasks for three different embedding methods used in the five pre-trained architectures. The numbers in the parenthesis denote the percent increment of accuracy compared to those of the last layer.

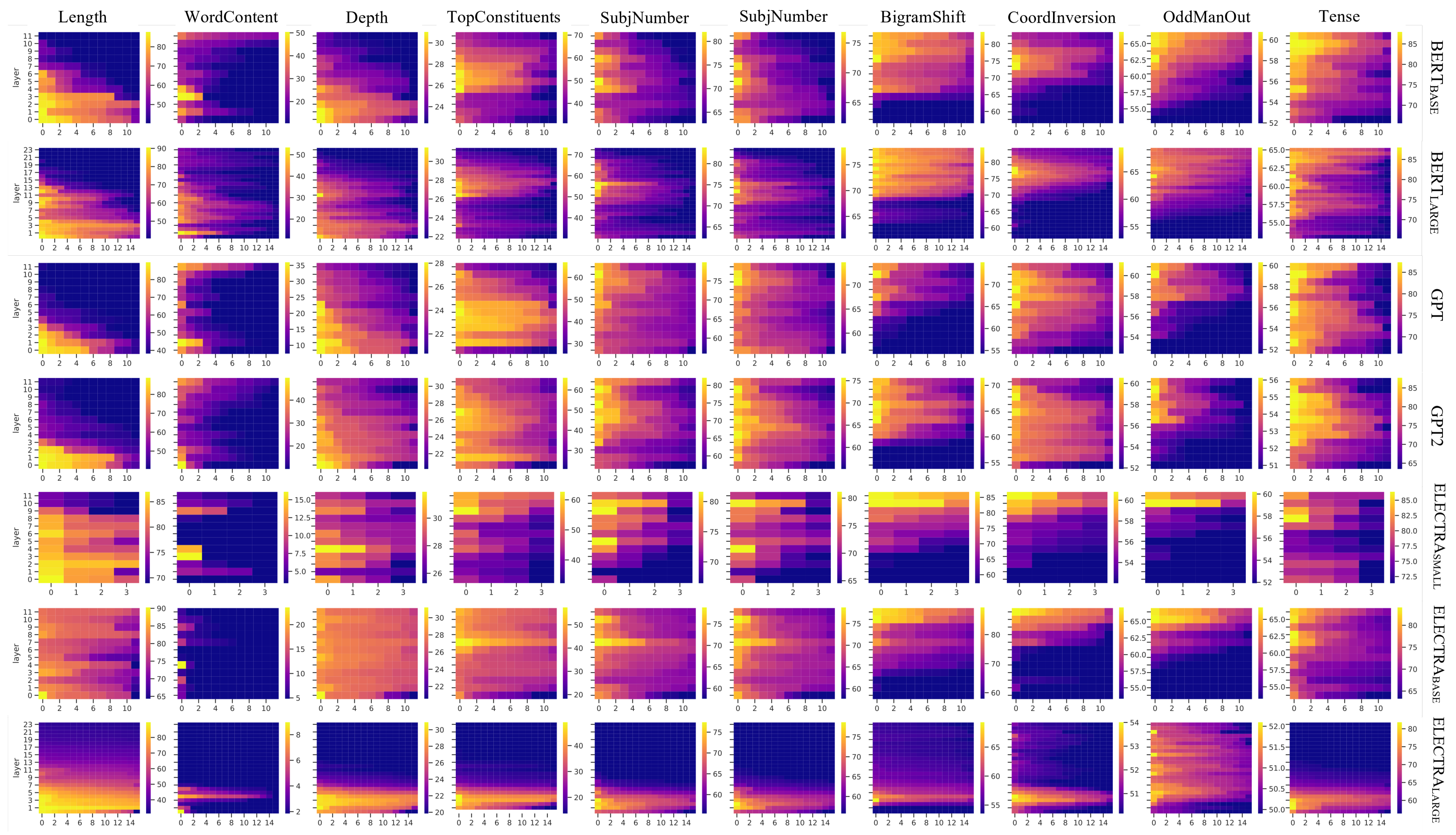

Figure 6: Heatmaps of the attention head-wise evaluation on the ten sentence probing tasks with BERT$_{\text{BASE}}$, BERT$_{\text{LARGE}}$, GPT, GPT2, ELECTRA$_{\text{SMALL}}$, ELECTRA$_{\text{BASE}}$, and ELECTRA$_{\text{LARGE}}$. 70 heat maps correspond to the ten tasks (Length, WordContent, Depth, TopConstituents, SubjNum, ObjNum, BigramShift, CoordInversion, OddManOut, and Tense from the left). In each heat map, x-axis and y-axis show the index values of the attention heads and the layer numbers (the lower, the closer to the initial input), respectively. The brighter the color, the higher the accuracy for the attention head and hence more important for the task. Note that the attention heads on the same layer are ordered by their classification accuracy values (i.e. an attention head with the highest accuracy on a layer is at the left-most location).

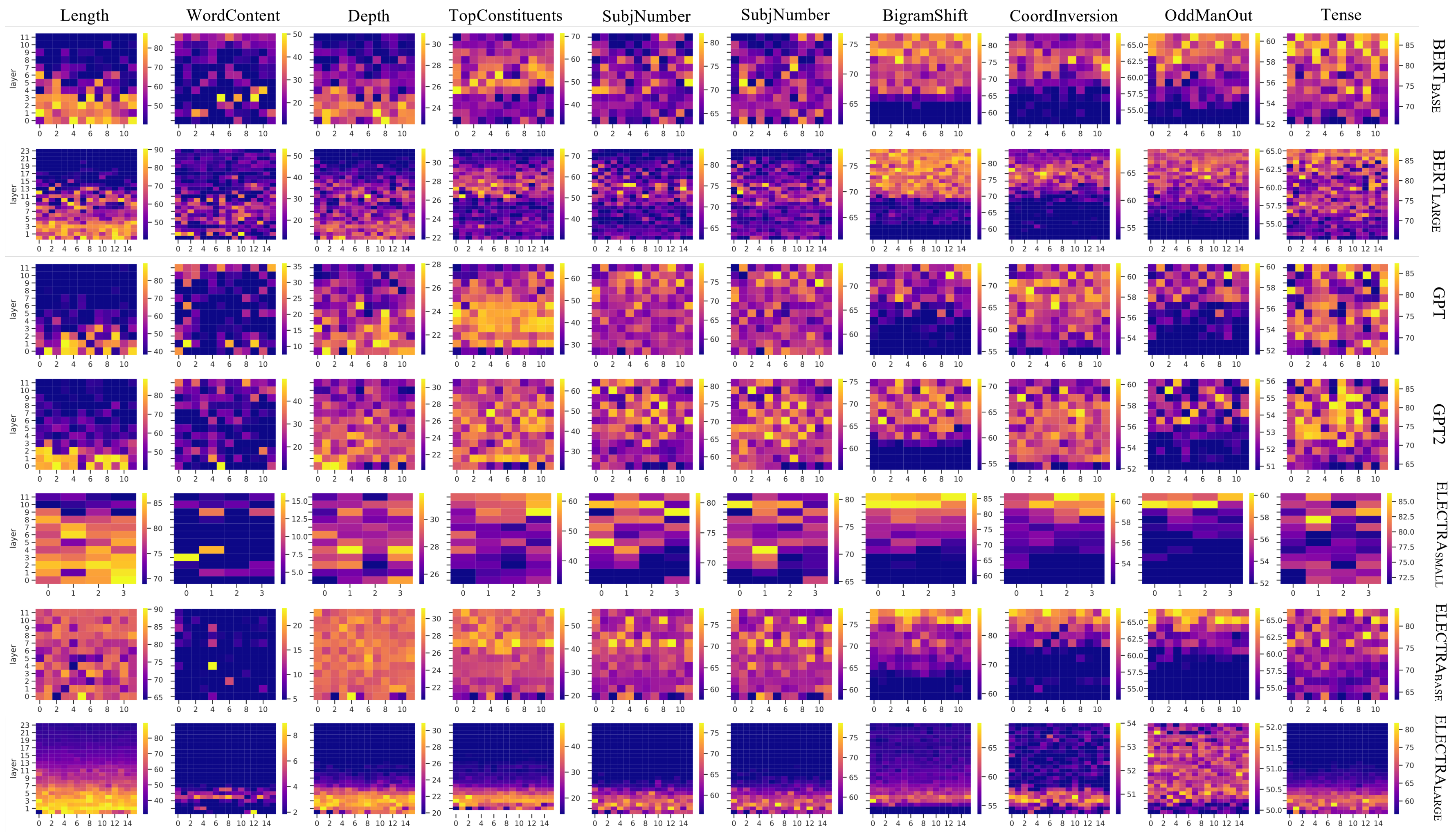

Figure 7: **Unsorted** heatmaps of the attention head-wise evaluation on the ten sentence probing tasks with $\text{BERT}_{\text{BASE}}$, $\text{BERT}_{\text{LARGE}}$, GPT, GPT2, $\text{ELECTRA}_{\text{SMALL}}$, $\text{ELECTRA}_{\text{BASE}}$, and $\text{ELECTRA}_{\text{LARGE}}$. 70 heat maps correspond to the ten tasks (Length, WordContent, Depth, TopConstituents, SubjNum, ObjNum, BigramShift, CoordInversion, OddManOut, and Tense from the left). In each heat map, x-axis and y-axis show the index values of the attention heads and the layer numbers (the lower, the closer to the initial input), respectively. The brighter the color, the higher the accuracy for the attention head and hence more important for the task. Note that the attention heads on the same layer are not ordered like Figure 6.