

Structured Self-Attention Weights Encode Semantics in Sentiment Analysis

Zhengxuan Wu¹, Thanh-Son Nguyen², Desmond C. Ong^{2,3}

¹Symbolic Systems Program, Stanford University

²Institute of High Performance Computing, Agency for Science, Technology and Research, Singapore

³Department of Information Systems and Analytics, National University of Singapore

wuzhengx@stanford.edu, Nguyen_Thanh_Son@ihpc.a-star.edu.sg,

dco@comp.nus.edu.sg

Abstract

Neural *attention*, especially the self-attention made popular by the Transformer, has become the workhorse of state-of-the-art natural language processing (NLP) models. Very recent work suggests that the self-attention in the Transformer encodes syntactic information; Here, we show that self-attention scores encode semantics by considering sentiment analysis tasks. In contrast to gradient-based feature attribution methods, we propose a simple and effective Layer-wise Attention Tracing (LAT) method to analyze structured attention weights. We apply our method to Transformer models trained on two tasks that have surface dissimilarities, but share common semantics—sentiment analysis of movie reviews and time-series valence prediction in life story narratives. Across both tasks, words with high aggregated attention weights were rich in emotional semantics, as quantitatively validated by an emotion lexicon labeled by human annotators. Our results show that structured attention weights encode rich semantics in sentiment analysis, and match human interpretations of semantics.

1 Introduction

In recent years, variants of neural network attention mechanisms such as local attention (Bahdanau et al., 2015; Luong et al., 2015) and self-attention in the Transformer (Vaswani et al., 2017) have become the *de facto* go-to neural models for a variety of NLP tasks including machine translation (Luong et al., 2015; Vaswani et al., 2017), syntactic parsing (Vinyals et al., 2015), and language modeling (Liu and Lapata, 2018; Dai et al., 2019).

Attention has brought about increased performance gains, but what do these values ‘mean’? Previous studies have visualized and shown how learnt attention contributes to decisions in tasks like natural language inference and aspect-level sentiment (Lin et al., 2017; Wang et al., 2016;

Ghaeini et al., 2018). Recent studies on the Transformer (Vaswani et al., 2017) have demonstrated that attention-based representations encode syntactic information (Tenney et al., 2019) such as anaphora (Voita et al., 2018; Goldberg, 2019), Parts-of-Speech (Vig and Belinkov, 2019) and dependencies (Raganato and Tiedemann, 2018; Hewitt and Manning, 2019; Clark et al., 2019). Other researchers have also done very recent extensive analyses on self-attention, by, for example, implementing gradient-based Layer-wise Relevance Propagation (LRP) method on the Transformer (Voita et al., 2019) to study attributions of gradient-scores to heads, or graph-based aggregation method to visualize attention flows (Abnar and Zuidema, 2020). These very recent works have not looked at whether the structured attention weights themselves aggregate on tokens with strong semantic meaning in tasks such as sentiment analysis. Thus, it is still unclear if the attention on input words may actually encode semantic information relevant to the task.

In this paper, we were interested in extending previous studies on attention and syntax further, by probing the structured attention weights and studying whether these weights encode task-relevant semantic information. In contrast to gradient-based attribution methods (Voita et al., 2019), we were explicitly interested in probing learnt attention weights rather than analyzing gradients. To do this, we propose a Layer-wise Attention Tracing (LAT) method to aggregate the structured attention weights learnt by self-attention layers onto input tokens. We show that these attention scores on input tokens correlate with an external measure of semantics across two tasks: a sentiment analysis task on a movie review dataset, and an emotion understanding task on a life stories narrative dataset. These tasks differ in structure (single-example classification vs. time-series regression), and in domain (movie reviews vs. daily life events), but should share the same

semantics, in that the same words should be important in both tasks. We propose a method of external validation of the semantics of these tasks, using emotion lexicons. We find evidence for the hypothesis that if self-attention mechanisms can learn emotion semantics, then LAT-calculated attention scores should be higher for words that have stronger emotional semantic meaning.¹

2 Attention-based Model Architecture

We use an encoder-decoder architecture as shown in Fig. 1. Our encoder is identical to the encoder of the Transformer (Vaswani et al., 2017), with an additional local attention layer (Luong et al., 2015). Our decoder is task-specific: a simple Multilayer Perceptron (MLP) for the classification task, and a LSTM followed by a MLP for the time-series prediction task.

Self-attention Layers. The encoder is identical to the original Transformer encoder and consists of a series of stacked self-attention layers. Each layer contains a multi-head self-attention layer, followed by an element-wise feed forward layer and residual connections. Following Vaswani et al. (2017), we use $L = 6$ stacked layers and $H = 8$ heads, and a hidden dimension of $D = 512$.

We briefly recap the Transformer equations, to better illustrate our LAT method, which traces attention back through the layers. For a given self-attention layer $l \in [1, L]$, we denote the input to l using $\mathbf{X}^l \in \mathbb{R}^{N \times D_e}$, which represents N tokens, each embedded using a D_e -dimensional embedding. We keep the same input embedding size for all layers. The first layer takes as input the word tokens. A self-attention layer learns a set of *Query*, *Key* and *Values* matrices that are indexed by l (i.e., weights are not shared across layers). Formally, these matrices are produced in parallel:

$$\mathbf{Q}^l = f_q^l(\mathbf{X}^l), \quad \mathbf{K}^l = f_k^l(\mathbf{X}^l), \quad \mathbf{V}^l = f_v^l(\mathbf{X}^l) \quad (1)$$

where $f_{\{q, k, v\}}^l(\cdot)$ are each parameterized by a linear layer, and each matrix is of size $N \times D$. To enable multi-head attention, \mathbf{Q} , \mathbf{K} and \mathbf{V} are partitioned into H separate $N \times D_h$ attention heads indexed by $h \in [1, H]$, where $D_h = \frac{D}{H} = 64$.

Each head learns a self-attention matrix $\alpha_h^{s(l)}$ using the scaled inner product of \mathbf{Q}_h and \mathbf{K}_h followed

¹Code is available at https://github.com/frankaging/LAT_for_Transformer

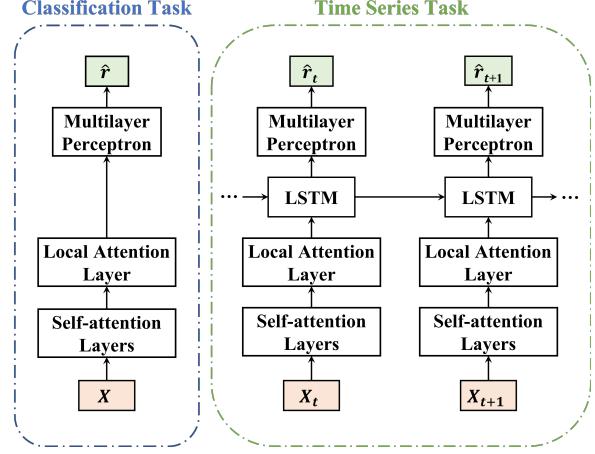


Figure 1: Attention-based encoder-decoder model architecture for classification task (left) and time-series task (right); The latter has a recurrent unit to generate predictions over time.

by a *softmax* operation. The self-attention matrix $\alpha_h^{s(l)}$ is then multiplied by \mathbf{V}_h to produce \mathbf{Z}_h^l :

$$\alpha_h^{s(l)} = \text{softmax}\left(\frac{\mathbf{Q}_h^l \mathbf{K}_h^{l T}}{\sqrt{D_h}}\right) \quad \in \mathbb{R}^{N \times N} \quad (2)$$

$$\mathbf{Z}_h^l = \alpha_h^{s(l)} \mathbf{V}_h^l \quad \in \mathbb{R}^{N \times D_h} \quad (3)$$

Next, we concatenate \mathbf{Z}_h^l from each head h to produce the output of layer l (i.e., the input to layer $l + 1$) \mathbf{X}^{l+1} ,

$$\mathbf{X}^{l+1} = f_\psi^l([\mathbf{Z}_1^l, \dots, \mathbf{Z}_H^l]) \quad \in \mathbb{R}^{N \times D_e} \quad (4)$$

where $f_\psi^l(\cdot)$ is parameterized by two fully connected feed-forward layers (with 64 dimensions for the first layer then scaling back to D_e -dimensions) with residual connections and layer normalization. \mathbf{X}^{l+1} is fed upwards to the next layer.

Local Attention Layer. The output from the last self-attention layer \mathbf{X}^{L+1} is fed into a local attention layer. We then take a weighted sum over row vectors of the output, and produces a context vector c using learnt local attention vector α^c :

$$\alpha^c = \text{softmax}\left(f_\phi(\mathbf{X}^{L+1})\right) \quad \in \mathbb{R}^N \quad (5)$$

$$c = (\mathbf{X}^{L+1})^T \alpha^c = \sum_{i=1}^N \alpha_i^c x_i^{L+1} \quad \in \mathbb{R}^{D_e} \quad (6)$$

where $f_\phi(\cdot)$ is parameterized by a multi-layer perceptron (MLP) with two hidden layers that are 128-dimensional and 64-dimensional. The MLP layers are trained with dropout of $p = 0.3$.

Decoder. For the classification task, the context vector c is fed into a decoder $f_{dec}(\cdot)$ parameterized by a MLP to produce the output label. For the time-series task, context vectors c_t from each time t are fed into a LSTM (Hochreiter and Schmidhuber, 1997) layer with 300-dimensional hidden states before passing through a MLP. Both the MLP for the classification and the time-series tasks have the same 64-dimensional hidden space, and are trained with dropout of $p = 0.3$. A complete model description can be found in the Appendix.

3 Layer-wise Attention Tracing

To study whether structured attention weights encode semantics, we propose a tracing method, Layer-wise Attention Tracing (LAT), to trace the attention ‘paid’ to input tokens (i.e. words) through the self-attention layers in our encoder. LAT, illustrated in Fig. 2, involves three main steps. First, starting from the local attention layer and a fixed “quantity” of attention, we distribute attention weights back to \mathbf{Z}_h^L , the last self-attention layer of each head $h \in [1, H]$. Second, we trace the attention back through each self-attention layer $l \in [1, L]$. Third, from the first layer of each head, we trace the attention back onto each token in the input sequence, by accumulating attention scores from each head to the corresponding position. We do not consider the decoder in LAT, as the MLP and LSTM layers in the decoder do not modify attention. Furthermore, we specifically ignore the feedforward layers and residual connections in the encoder, as we were interested in the attention α , not the neural activations they modify—this is our main differentiation from gradient-based or relevance-based work (Voita et al., 2019), and we note another recent paper (Abnar and Zuidema, 2020) that made the same assumptions.

Tracing Local Attention. Given an input sequence X of length N tokens, the forward pass of the model (Eqn. 1-6) transforms X into the context vector c . We consider how a fixed quantity of attention, \mathbf{A}^c , gets divided back to the various heads. We refer to this quantity as the *Attention Score* that is accumulated down through the layers. From Eqn. 4 and Eqn. 6, we note that c is a function of concatenated \mathbf{Z}^L from the last self-attention layer, from each of the heads:

$$c = \sum_{i=1}^N \alpha_i^c f_\psi^L([z_{1(i)}^L, \dots, z_{H(i)}^L]) \quad (7)$$

where $z_{h(i)}^L$ is the attended *Value* vector from head $h \in [1, H]$ of the last layer L at position $i \in [1, N]$. On the forward pass, the contribution of head h at position i , $z_{h(i)}^L$, is weighted by α_i^c ; Thus, on this first step of LAT, we divide the attention score \mathbf{A}^c back to head h at position i , using α_i^c :

$$\mathbf{A}_{h(i)}^{L+1} = \alpha_i^c \mathbf{A}^c \quad (8)$$

We use this notation to allude that this is the attention weights coming down from the “($L + 1$)-th layer”, to follow the logic of the next step of LAT. Without loss of generality, we can set the initial attention score at the top, \mathbf{A}^c , to be 1, then all subsequent attention scores can be interpreted as a proportion of the initial attention score. Note that in our attention tracing, we are interested in accumulating the attention $\mathbf{A}_{h(i)}^l$ for each layer $l \in [1, L]$ at each position i , and so we focus on the attention weights (and not the hidden states that the attention multiplies, \mathbf{Z}_h or \mathbf{V}_h), which remain unchanged through f_ψ .

Tracing Self Attention. On the forward pass, Eqn. 3 applies the self-attention weights. We rewrite this equation to make the indices explicit:

$$z_{h(j)}^l = \sum_{i=1}^N \alpha_{h(i \rightarrow j)}^{s(l)} v_{h(i)}^l \quad (9)$$

where $v_{h(i)}^l$ denotes the i -th row of \mathbf{V}_h^l (i.e., corresponding to the token in position i), and $\alpha_{h(i \rightarrow j)}^{s(l)}$ is the (j, i) element of $\alpha_h^{s(l)}$, such that it captures the attention from position i to position j . The attended values \mathbf{Z}_h^l then undergo two sets of feed-forward layers: Eqn. 4 with f_ψ^l to get \mathbf{X}^l and Eqn. 1 with f_v^l to get \mathbf{V}_h^{l+1} .

Using $\mathbf{A}_{h(j)}^l$ to denote the attention score accumulated at head h , position j , layer l , we can trace the attention coming down from the next-higher layer based on Eqn. 9:

$$\mathbf{A}_{h(i)}^l = \sum_{j=1}^N \alpha_{h(i \rightarrow j)}^{s(l)} \mathbf{A}_{h(j)}^{l+1} \quad (10)$$

To confirm our intuition, on the forward pass (see Eqn. 9 and Fig. 2), to get the hidden value at position j on the “upper” part of the layer, we sum $\alpha_{h(i \rightarrow j)}^{s(l)}$ over i (the indices of the “lower” layer). Thus, on the LAT pass downwards (Eqn. 10), to get $\mathbf{A}_{h(i)}^l$ as position i on the “lower” layer, we sum the corresponding $\alpha_{h(i \rightarrow j)}^{s(l)}$ ’s over j .

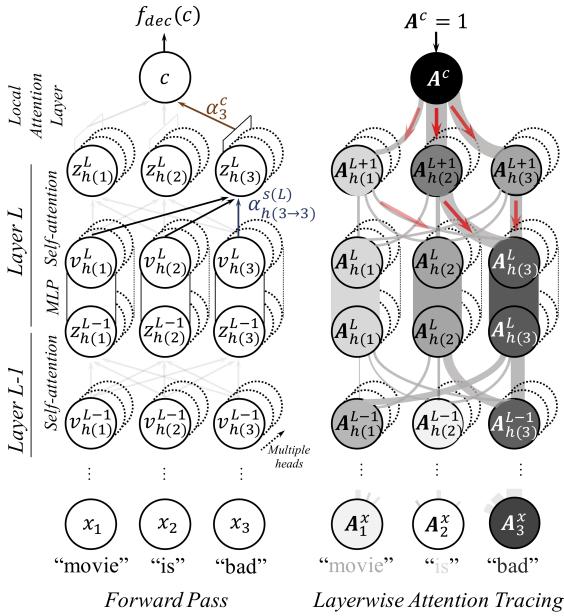


Figure 2: An illustration of the Layer-wise Attention Tracing (LAT) method with an example forward pass through head h . Left: On the forward pass, learnt attention weights are represented by lines producing Z_h^l from values V_h^l via self-attention (Eqn. 2-3) and the context vector c from the last layer via local attention (Eqn. 5). Dashed circles represents multiple heads, and vertical columns represent MLP transformations, which do not redistribution attention. Right: LAT on a ‘backward pass’. The thickness of the edges represents accumulating attention. Attention from incoming edges are accumulated at each position in each layer, as in Eqn. 10. Darker colors maps to greater accumulated attention scores. In this example, the input token “bad” receives the highest attention score.

Tracing to input tokens. Finally, for each input token X_i , we sum up the attention weights from each head at the corresponding position in the first layer to obtain the accumulated attention weights paid to token X_i :

$$A_i^x = \sum_{h=1}^H A_{h(i)}^1 \quad (11)$$

In summary, Eqns. 8, 10, and 11 describe the LAT method for tracing through the local and self-attention layers back to the input tokens X_i .

4 Related Work

There has been extensive debate over what attention mechanisms learn. On the one hand, researchers have developed methods to probe learnt self-attention in Transformer-based models, and show that attention scores learnt by models like

BERT encode syntactic information like Parts-of-Speech (Vig and Belinkov, 2019), dependencies (Hewitt and Manning, 2019; Raganato and Tiedemann, 2018), anaphora (Goldberg, 2019; Voita et al., 2018) and other parts of the traditional NLP pipeline (Tenney et al., 2019). These studies collectively suggest that self-attention mechanisms learn to encode syntactic information, which led us to propose the current work on whether self-attention can similarly learn to encode *semantics*.

On the other hand, there are also other papers questioning the interpretations the field has placed on attention. These researchers show that attention weights have a low correlation with gradient-based measures of importance (Jain and Wallace, 2019; Serrano and Smith, 2019; Vashishth et al., 2019). More recent analysis suggest that in certain regimes for the Transformer (i.e., sequence length greater than attention head dimension²), attention distributions are non-identifiable, posing problems for interpretability (Brunner et al., 2020). In our work, we provide a method that can trace attention scores in Transformers to the input tokens, and show with both qualitative and quantitative evidence that these scores are semantically meaningful.

Beyond attention-based studies, there have been numerous studies that proposed gradient-based attribution analyses (Dimopoulos et al., 1995; Gevrey et al., 2003; Simonyan et al., 2013) and layer-wise relevance propagation (Bach et al., 2015; Li et al., 2016; Arras et al., 2017). Most related to the current work is Voita et al. (2019), who extended layer-wise relevance propagation to the Transformer to examine the contribution of individual heads to the final decision. In parallel, Abnar and Zuidema (2020) recently proposed a method to roll-out structured attention weights inside the Transformer model, which is similar to our LAT method we propose here, although we provided more analysis via an external validation using external knowledge. We sought to investigate the attention accumulated onto individual input tokens using attention tracing, in a more similar manner to Vig and Belinkov (2019) for syntax or how Voita et al. (2018) looked at the attention paid to other words. We also calculate a gradient-based score (see Eqn. 13) to contrast our attention results with, and though these two scores are correlated (see Footnote 6), they behave differently in our analyses.

²We note that our models do not fall into this regime.

Model	SST-5
	Accuracy (SD ^{runs})
RNTN (Socher et al., 2013)	45.7 (-)
BiLSTM (Tai et al., 2015)	46.5 (-)
Transformer + Position encoding (Ambartsoumian and Popowich, 2018)	45.0 (0.4)
DiSAN (Shen et al., 2018)	51.7 (-)
<i>Our Self-attention</i>	47.5 (0.2)
SEND	
CCC (SD ^{runs} , SD ^{eg})	
LSTM (Ong et al., 2019)	.40 (-, .32)
SFT (Wu et al., 2019)	.34 (-, .33)
Human (Ong et al., 2019)	.50 (-, .12)
<i>Our Self-attention + LSTM</i>	.54 (.02, .36)

Table 1: Summary of results. Top: Test accuracy averages and standard deviations (in brackets) for SST-5. Bottom: Test CCC averages and standard deviations on the SEND Test set. We additional calculate a SD^{eg} over the CCCs of the same model over the (39) examples in the Test set, as used in previous papers, to better estimate generalizability to new unseen examples.

5 Datasets

To show that our interpretation methods can generalize across different types of datasets, we apply our method to two tasks with different characteristics, namely, sentiment classification of movie reviews on the Stanford Sentiment Treebank (SST), and time-series valence regression over long sequences narrative stories on the Stanford Emotional Narratives Dataset (SEND).

5.1 Stanford Sentiment Treebank

We used the fine-grained (5-class) version of the Stanford Sentiment Treebank (SST-5) movie review dataset (Socher et al., 2013), which has been used in previous studies of interpretability of neural network models (Li et al., 2016; Arras et al., 2017). All sentences³ were tokenized, and preprocessed by lowercasing, similar to (Li et al., 2016). We embed each token using 300-dimensional GloVe word embeddings (Pennington et al., 2014). Each sentence is labeled via crowdsourcing with one of five sentiment classes {Very Negative, Negative, Neutral, Positive, and Very Positive}. We used

³Although the SST contains labels on each parse tree of the reviews, we only considered full sentences.

the same dataset partitions as in the original paper: a *Train set* (8544 sentences, average length 19 tokens), a *Validation set* (1101 sentences, average length 19 tokens) and a *Test set* (2210 sentences, average length 19 tokens). Models are trained to maximize the 5-class classification accuracy by minimizing multi-class cross-entropy loss. We compare our model with previous works on SST that are based on LSTM (Tai et al., 2015) and Transformer (Ambartsoumian and Popowich, 2018; Shen et al., 2018).

5.2 Stanford Emotional Narratives Dataset

The SEND (Ong et al., 2019) comprises videos of participants narrating emotional life events. Each video is professionally transcribed, and annotated via crowdsourcing with emotion valence scores ranging from “Very Negative” [-1] to “Very Positive” [1] continuously sampled at every 0.5s. Details can be found on the authors’ GitHub repository. The SEND has previously been used to train deep learning models to predict emotion valence over time (Ong et al., 2019; Wu et al., 2019).

The SEND has 193 transcripts, and each one contains multiple sentences. We preprocess them by tokenizing and lowercasing as in (Ong et al., 2019; Wu et al., 2019). Additionally, we divide each transcript into 5-second time windows by using timestamps provided in the dataset. We use the average valence scores during a time window as the label of that window. We use the same partitions as in the original paper: a *Train set* (114 transcripts, average length 357 tokens, average window length 13 tokens), a *Validation set* (40 transcripts, average length 387 tokens, average window length 15 tokens) and a *Test set* (39 transcripts, average length 333 tokens, average window length 13 tokens). We embed each token in the same way as for SST-5. As in the original papers (Ong et al., 2019; Wu et al., 2019), we use the Concordance Correlation Coefficient (CCC (Lin, 1989)) as our evaluation metric (See Appendix for the definiton). We compare our model with previous works on SEND that use LSTM (Ong et al., 2019) and Transformer (Wu et al., 2019).

6 Results and Analysis

6.1 Model training and results

We report the results of our Transformer-based models in Table 1 with performances of state-of-the-art (SOTA) models trained with these two datasets.

We selected models in the literature that are the most representative and relevant to our models. Our Transformer-based model for the SST-5 classification task (Fig. 1) achieves good performance, with an accuracy (\pm standard deviation) of $47.5\% \pm 49.9\%$ on the five-class sentiment classification. For the SEND dataset, our model outperforms previous SOTA models and even average human performance on this task, with a mean CCC of $.54 \pm .36$ on the Test set. Interestingly, our window-based Transformer encoder increases performance compared to the Simple Fusion Transformer proposed by Wu et al. (2019), who used a Transformer-based encoder over the whole narrative sequence.

Both models are trained with the *Adam* (Kingma and Ba, 2015) optimization algorithm with a learning rate of 10^{-4} . As our goal was analyzing structured attention weights, not maximizing performance, we manually specified hyperparameters without any grid search. We include details about our experiment setup in the Appendix.

Given that our Transformer-based models achieved comparable state-of-the-art performance on the SST and SEND, we then proceed to analyze the attention scores produced by LAT on these models. After computing A_i^x for all the words in a given sequence, we normalize attention scores using the softmax function to have them sum to 1.

6.2 Visualizing Layerwise Attention Tracing

The *flow diagram* in Fig. 3 visualizes how attention aggregates using LAT across all heads and layers for the model trained with SST-5 for an example input. Rows represent self-attention layers and columns represent attention heads. Dots represent different tokens at head $h \in [1, H]$ (left to right), position $i \in [1, N]$ of layer $l \in [1, L]$ (bottom to top). Dots in the bottom-most layer represents input tokens. The darker the color of each dot, the higher the accumulated attention score at that position, calculated using by Eqns. 8, 10 and 11. Attention weights $a_{h(i \rightarrow j)}^{s(l)}$ in each layer are illustrated by lines connecting tokens in consecutive layers.

This diagram illustrates some coarse-grained differences between heads. For example, all heads in the top last layer distributed attention fairly equally across all tokens. Other heads (e.g., Head 6, Layer 4, and Head 8, Layer 3) have a downward-triangle pattern, where attention weights are accumulated to a specific token in a lower-layer, while others (e.g. Head 5, Layer 1) seem to re-distribute accumulated

attention more broadly. Finally, at the input layer, we note that attention scores seem to be highest for words with strong emotion semantics.

6.3 Sentiment Representations of Words

To validate that the attention weights aggregated on the input tokens by LAT is semantically meaningful, we rank all unique word-level tokens in the Test set by their averaged attention scores received from all sequences that they appear. Concretely, we first use LAT to trace attention weights paid to input tokens for every sequences in the Test set. For tokens that appear more than once, we average their attention scores across occurrences. We then rank tokens by their average attention score, and illustrate in Fig. 4 using word clouds where a larger font size maps to a higher average attention score. For both datasets, we observe that words expressing strong emotions also have higher attention scores, see e.g. *sorry*, *painful*, *unsatisfying* for SST-5, and *congratulations*, *freaking*, *comfortable* for SEND. We note that stop words do not receive high attention scores in either of the datasets.

6.4 Quantitative validation with an emotion lexicon

One advantage of extracting emotion semantics from natural language text is that the field has amassed large, annotated references of emotion semantics. We refer, of course, to the emotion lexicons that earlier NLP researchers used for sentiment analysis and related tasks (Hu and Liu, 2004). Although they seem to have fallen out of favor with the rise of deep learning (and the hypothesis that deep learning can learn such knowledge in a data-driven manner), in our task, we sought to use emotion lexicons as an external validation of what our model learns.

We used a lexicon (Warriner et al., 2013) of nearly 14,000 English lemmas that are each annotated by an average of 20 volunteers for emotional valence, which corresponds exactly to the semantics in our tasks. The mean valence ratings in this lexicon are real-valued numbers from 1 to 9.

We hypothesize that our LAT method produce attention scores such that words having higher scores will tend to have greater emotional meaning. Additionally, since our attention scores do not differentiate emotion “directions” (i.e., negative and positive), these attention scores should be high for both very positive words, *as well as* very negative words. Thus, we expect a *U-shaped* relationship

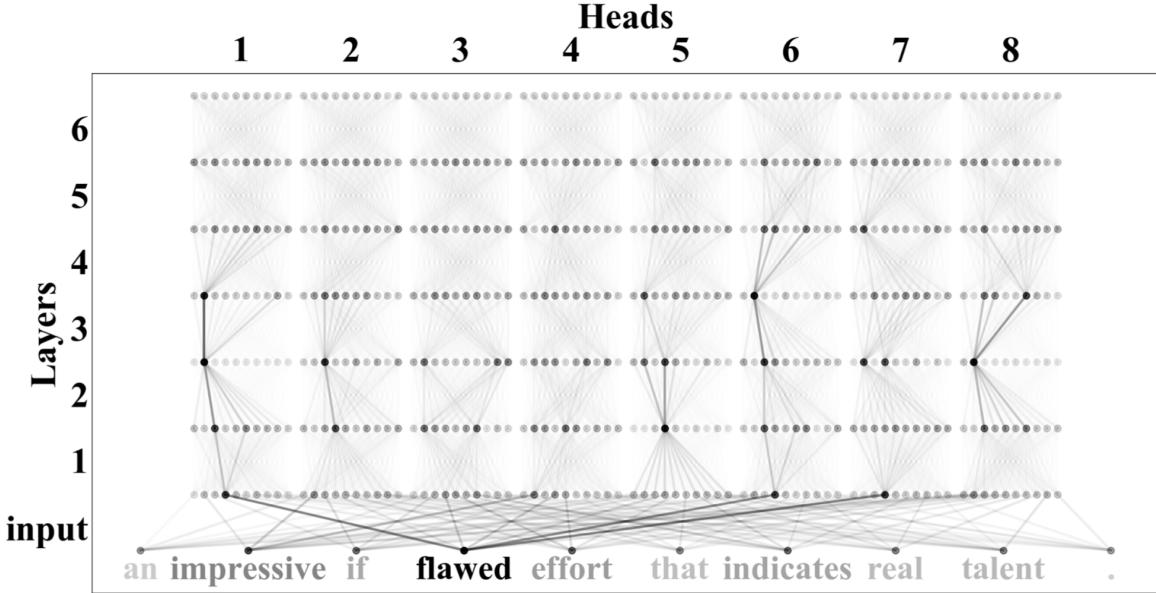


Figure 3: An example *flow diagram* of attention distributed through self-attention layers in action. On the bottom, the font weights illustrate the accumulated attention weights paid to a particular word. The predicted label and the true label are both *positive*. Note that the color of the dots represent the attention weights $A_{h(i)}^l$ (Eqn. 10), not the activation of those neurons, and so these are not affected by the states that are shared across heads.

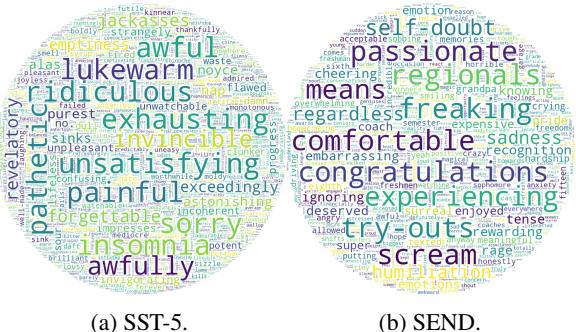


Figure 4: Word cloud created based on averaged accumulated attention weights assigned to words in the vocabularies of Test sets.

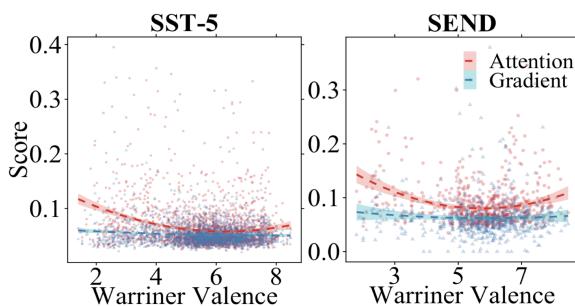


Figure 5: Scatterplot shows scores on y-axis derived from Eqn. 12 (LAT attention scores A_w in red circles and gradient scores G_w in blue triangles) and corresponding emotional valence ratings, Val_w , from the Warriner et al. (2013) lexicon on x-axis. Shared vocabulary size is 2335 for SST-5, and 660 for SEND.

between our attention scores and the lexicon’s valence ratings. We examine this hypothesis by fitting a quadratic regression equation⁴:

$$A_w = b_0 + b_1 Val_w + b_2 [Val_w]^2 + \epsilon \quad (12)$$

where A_w is the averaged attention score of a particular word w derived by the LAT method, and Val_w represents the valence rating of that word from the Warriner et al. (2013) lexicon. We hypothesized a statistically-significant coefficient b_2 on the quadratic term.

To contrast our attention score with another measure of importance, the gradient, i.e., how important the inputs are to affecting the output (Li et al., 2016), we also calculate a gradient score on each token by computing squared partial derivatives:

$$G_{w(d)} = \left(\frac{\partial f_\xi}{\partial w_{(d)}}(w) \right)^2 \quad (13)$$

where f_ξ can be parameterized by neural networks, and $G_{w(d)}$ is the gradient of a particular space dimension d of the embedding for the input token w . We then regress G_w on the lexicon valence ratings using Eqn. 12.

We plot both our attention scores and gradient scores for each word against Warriner et al.

⁴Specifically, we used the following formula in R syntax: `lm(att ~ poly(val, 2))`, where `poly()` creates orthogonal polynomials to avoid collinearity issues.

		Heads								
		rand	1	2	3	4	5	6	7	8
SST-5	v.neg	.07	.15	.10	.09	.17	.12	.15	.09	.12
	v.pos	.02	.12	.09	.06	.15	.10	.13	.06	.11
	both	.05	.20	.10	.15	.18	.14	.17	.15	.12
SEND	v.neg	.13	.16	.17	.17	.22	.20	.16	.16	.20
	v.pos	.03	.14	.14	.14	.16	.17	.14	.13	.17
	both	.11	.14	.16	.15	.24	.17	.12	.15	.16

Figure 6: Heatmap of proportions of attention paid to words with selected semantics tags. The leftmost column ‘rand’ shows the proportions if attention weights are uniformly distributed at chance.

(2013) valence ratings, in Fig. 5. For both tasks, we considered only words that appeared in both our Test sets and the lexicon, and plot only scores below 0.4 to make the plot more readable⁵. We can see clearly that there exist a U-shaped, quadratic relationship between attention scores and the Warriner valence ratings ($b_2 = 0.283, SE = 0.040, t = 7.04, p < .001$ for SST-5; $b_2 = 0.242, SE = 0.039, t = 6.21, p < .001$ for SEND). Our results support our hypothesis that the attention scores recovered by our LAT method do track emotional semantics. As a result, we show that structured attention weights may encode semantics independent of other types of connections in the model (e.g., linear feedforward layers and residual layers.). By contrast, there is no clear quadratic relationship between gradient scores and valence ratings across both tasks (SST-5, $p = 0.19$; SEND, $p = 0.28$)⁶.

6.5 Head Attention on Sentiment Words

We next analyze the amount of attention paid to sentiment words in each head. Within each head h , we analyze the proportion of accumulated attention $A_{h(i)}^1$ on emotional words, specifically focusing on *very positive* and *very negative* words⁷, aggregated

⁵This plotting rule only filtered out less than 1% of words in the Test sets: .171% for SST-5 and .754% for SEND.

⁶On the SST, A_i^x and $G_{i(d)}^x$ are correlated at $\rho = .80$, and on the SEND, $\rho = .37$. The two values are highly correlated (on the SST), but vary differently with respect to valence.

⁷For SST-5, we used the original word-level *very positive* and *very negative* labels in the dataset. For SEND, we used the Warriner lexicon and chose a cutoff ≥ 6.5 for very positive, and < 3.5 for very negative.

over the Test sets:

$$p_{A_h^1}(\text{tag}) = \frac{\sum_{X \in \mathcal{X}} \sum_{i=1}^{|X|} (A_{h(i)}^1) \mathbb{1}_{\text{label}(x_i)=\text{tag}}}{\sum_{X \in \mathcal{X}} \sum_{i=1}^{|X|} (A_{h(i)}^1)} \quad (14)$$

where \mathcal{X} is the subset of sequences that contain at least 1 word with the selected tag⁸.

Fig. 6 shows the proportion of attention accumulated by heads to *very positive* and *very negative* words, compared with chance. All heads do seem to pay more attention to strongly emotional words, compared to chance, and some heads seem to ‘specialize’ more: For example, Head 4 in our SEND model pays 24% of its accumulated attention to *very negative* words while the mean of all other heads is closer to 15%. While Fig. 6 is specific to the model we trained, it is illustrative that specialization to strong emotional semantics does emerge from the learnt attention weights.

7 Discussion

In this work, we analyzed whether structured attention weights encode semantics in sentiment analysis tasks, using our proposed probing method LAT to trace attention through multiple layers in the Transformer. We demonstrated that the accumulated attention scores tended to favor words with greater semantic meaning, in this case, emotional meaning. We applied LAT to two tasks having similar semantics, and show that our results generalize across both tasks/domains. We validated our results quantitatively with an emotion lexicon, and showed that our attention scores are highest for both highly positive and highly negative words—our *a priori* hypothesis for the quadratic, “U-shaped” relationship. We also found some evidence for specialization of heads to emotional meaning. Although it may seem that our attention tracing is “incomplete” as it does not take into account the feed-forward layers and residual connections, by contrast, this quadratic relationship was not shown by pure gradient-based importance, which suggests that there may be some utility to looking only at attention.

We believe that attention in its various forms (Luong et al., 2015; Vaswani et al., 2017) are not only effective for performance, but may also provide

⁸That is, when calculating $p_{A_h^1}$ (*very positive*), we exclude sequences that do not contain at least 1 *very positive* word.

interpretable explanations of model behaviour. It may not happen with today’s implementations; we may need to engineer inductive biases to constrain attention mechanisms in order to address issues of identifiability that Jain and Wallace (2019) and others have pointed out. And perhaps, attention should not be interpreted like gradient-based measures (see Fig. 5). This debate is not yet resolved, and we hope our contributions will be useful in informing future work on this topic.

References

- Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, Online. Association for Computational Linguistics.
- Artaches Ambartsoumian and Fred Popowich. 2018. Self-attention: A better building block for sentiment analysis neural network classifiers. In *Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 130–139.
- Leila Arras, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2017. Explaining recurrent neural network predictions in sentiment analysis. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 159–168.
- Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klaušen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7).
- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 4th International Conference on Learning Representations (ICLR)*.
- Gino Brunner, Yang Liu, Damian Pascual Ortiz, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2020. On identifiability in transformers. In *International Conference on Learning Representations (ICLR)*.
- Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. 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*, pages 276–286.
- Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988.
- Yannis Dimopoulos, Paul Bourret, and Sovan Lek. 1995. Use of some sensitivity criteria for choosing networks with good generalization ability. *Neural Processing Letters*, 2(6):1–4.
- Muriel Gevrey, Ioannis Dimopoulos, and Sovan Lek. 2003. Review and comparison of methods to study the contribution of variables in artificial neural network models. *Ecological modelling*, 160(3):249–264.
- Reza Ghaeini, Xiaoli Z Fern, and Prasad Tadepalli. 2018. Interpreting recurrent and attention-based neural models: A case study on natural language inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4952–4957.
- Yoav Goldberg. 2019. Assessing BERT’s syntactic abilities. *arXiv preprint arXiv:1901.05287*.
- John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. 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 4129–4138.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.
- Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge discovery and Data Mining*, pages 168–177.
- Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3543–3556.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015*.
- Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.
- Lawrence I-Kuei Lin. 1989. A concordance correlation coefficient to evaluate reproducibility. *Biometrics*, pages 255–268.

- Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*.
- Yang Liu and Mirella Lapata. 2018. Learning structured text representations. *Transactions of the Association for Computational Linguistics*, 6:63–75.
- Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.
- Desmond Ong, Zhengxuan Wu, Zhi-Xuan Tan, Marianne Reddan, Isabella Kahhale, Alison Mattek, and Jamil Zaki. 2019. Modeling emotion in complex stories: the Stanford Emotional Narratives Dataset. *IEEE Transactions on Affective Computing*.
- Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.
- 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.
- Sofia Serrano and Noah A Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951.
- Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. DiSAN: Directional Self-Attention Network for RNN/CNN-free language understanding. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y 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.
- Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 2358–2367.
- Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601.
- Shikhar Vashisht, Shyam Upadhyay, Gaurav Singh Tomar, and Manaal Faruqui. 2019. Attention interpretability across NLP tasks. *arXiv preprint arXiv:1909.11218*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.
- Jesse Vig and Yonatan Belinkov. 2019. Analyzing the structure of attention in a transformer language model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 63–76.
- Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems*, pages 2773–2781.
- Elena Voita, Pavel Serdyukov, Rico Sennrich, and Ivan Titov. 2018. Context-aware neural machine translation learns anaphora resolution. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 1264–1274.
- 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.
- Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615.
- Amy Beth Warriner, Victor Kuperman, and Marc Brysbaert. 2013. Norms of valence, arousal, and dominance for 13,915 English lemmas. *Behavior Research Methods*, 45(4):1191–1207.
- Zhengxuan Wu, Xiyu Zhang, Tan Zhi-Xuan, Jamil Zaki, and Desmond C Ong. 2019. Attending to emotional narratives. In *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 648–654. IEEE.

8 Supplementary Materials

8.1 Evaluation Metrics

Concordance Correlation Coefficient (CCC (Lin, 1989)): The CCC of vectors X and Y is:

$$\text{CCC}(X, Y) \equiv \frac{2\text{Corr}(X, Y)\sigma_X\sigma_Y}{\sigma_X^2 + \sigma_Y^2 + (\mu_X - \mu_Y)^2} \quad (15)$$

where $\text{Corr}(X, Y) \equiv \text{cov}(X, Y)/(\sigma_X\sigma_Y)$ is the Pearson correlation coefficient, and μ and σ denotes the mean and standard deviation respectively.

8.2 Experiment Setup

Computing Infrastructure: To train our models, we use a single Standard NV6 instance on Microsoft Azure. The instance is equipped with a single NVIDIA Tesla M60 GPU.

Average Runtime: With the computing infrastructure, it takes about 1.5 hrs to train both models, where each model is trained with 200 epochs. Both model reaches maximum performances in about 1.5 hrs at about 100 epochs.

Number of Trainable Parameters: The model trained on SST-5 that uses a LSTM decoder has 3,993,222 parameters. Additionally, the model trained on SEND that uses a MLP decoder has 4,715,362 parameters.

8.3 Task-specific Decoders

Long-Short Term Memory Network (LSTM): For the time-series task, we use a LSTM layer (Hochreiter and Schmidhuber, 1997) to decode the context vector c_i from our encoder for each window i to output a hidden vector h_i . Then, the hidden vector passes through a MLP to make the valence prediction:

$$h_i = \text{LSTM}(h_{i-1}, c_i) \quad (16)$$

$$\hat{r}_i = \text{MLP}(h_i) \quad (17)$$

Multilayer Perceptron (MLP): Our MLP contains 3 consecutive linear layers with a single ReLU activation in between layers. For the classification task, we feed in the context vector c from our encoder to MLP to make the sentiment prediction:

$$f_1(c) = \text{ReLU}(\mathbf{W}_1c + \mathbf{b}_1) \quad (18)$$

$$\hat{r}_i = \mathbf{W}_3 [\text{ReLU}(\mathbf{W}_2f_1(c) + \mathbf{b}_2)] + \mathbf{b}_3 \quad (19)$$

where $\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3, \mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$ are learnable parameters for linear layers. For the time-series task, the hidden vector h_t is the input instead.