Visualizing Attention in Transformer-Based Language Representation Models

Jesse Vig
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94304
`jesse.vig@parc.com`

Abstract

We present an open-source tool for visualizing multi-head self-attention in Transformer-based language representation models. The tool extends earlier work by visualizing attention at three levels of granularity: the attention-head level, the model level, and the neuron level. We describe how each of these views can help to interpret the model, and we demonstrate the tool on the BERT model and the OpenAI GPT-2 model. We also present three use cases for analyzing GPT-2: detecting model bias, identifying recurring patterns, and linking neurons to model behavior.

1 Introduction

In 2018, the BERT (Bidirectional Encoder Representations from Transformers) language representation model achieved state-of-the-art performance across NLP tasks ranging from sentiment analysis to question answering (Devlin et al., 2018). Recently, the OpenAI GPT-2 (Generative Pretrained Transformer-2) model achieved state-of-the-art results across several language modeling benchmarks in a zero-shot setting (Radford et al., 2019).

Underlying BERT and GPT-2 is the Transformer model, which uses a multi-head self-attention architecture (Vaswani et al., 2017a). An advantage of using attention is that it can help interpret a model's decisions by showing how the model attends to different parts of the input (Bahdanau et al., 2015; Belinkov and Glass, 2019). Various tools have been developed to visualize attention in NLP models, ranging from attention-matrix heatmaps (Bahdanau et al., 2015; Rush et al., 2015; Rocktäschel et al., 2016) to bipartite graph representations (Liu et al., 2018; Lee et al., 2017; Strobelt et al., 2018). A visualization tool designed specifically for the multi-head self-attention in the Transformer (Jones, 2017) was in-

troduced in (Vaswani et al., 2017b) and released in the Tensor2Tensor repository (Vaswani et al., 2018).

In this paper, we introduce a tool for visualizing attention in Transformer-based language representation models, building on the work of (Jones, 2017). We extend the existing tool in two ways: (1) we adapt it from the original encoder-decoder implementation to the decoder-only GPT-2 model and the encoder-only BERT model, and (2) we add two visualizations: the *model view*, which visualizes all of the layers and attention heads in a single interface, and the *neuron view*, which shows how individual neurons influence attention scores.

2 Visualization Tool

We present an open-source[1] tool for visualizing multi-head self-attention in Transformer-based language representation models. The tool comprises three views: an attention-head view, a model view, and a neuron view. Below, we describe these views and demonstrate them on the OpenAI GPT-2 and BERT models. We also present three use cases for GPT-2 showing how the tool might provide insights on how to adjust or improve the model. A video demonstration of the tool is available at `https://youtu.be/187JyiA4pyk`.

2.1 Attention-head view

The *attention-head view* visualizes the attention patterns produced by one or more attention heads in a given transformer layer, as shown in Figure 1 (GPT-2[2]) and Figure 2 (BERT[3]). In this view, self-attention is represented as lines connecting the tokens that are attending (left) with the tokens being

---

[1] `https://github.com/jessevig/bertviz`
[2] *GPT-2 small* pretrained model.
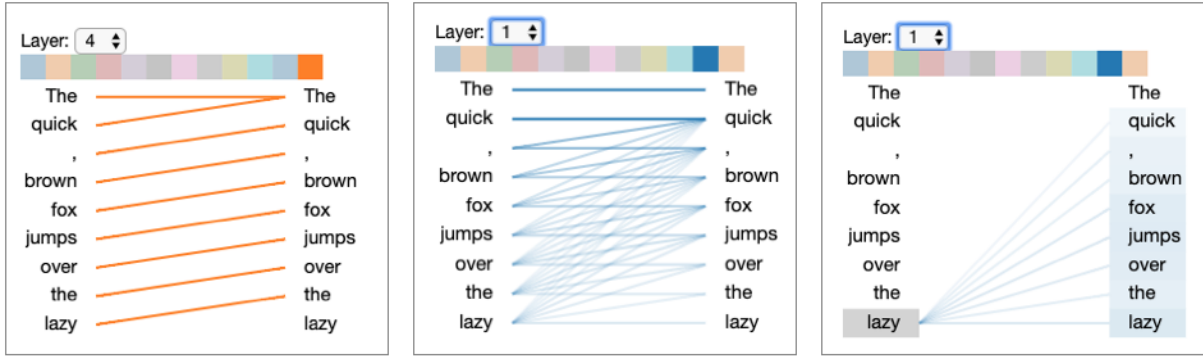[3] $BERT_{BASE}$ pretrained model.

Figure 1: Attention head view for GPT-2, for the input text *The quick, brown fox jumps over the lazy*. The left and center figures represent different layers / attention heads. The right figure depicts the same layer/head as the center figure, but with the token *lazy* selected.
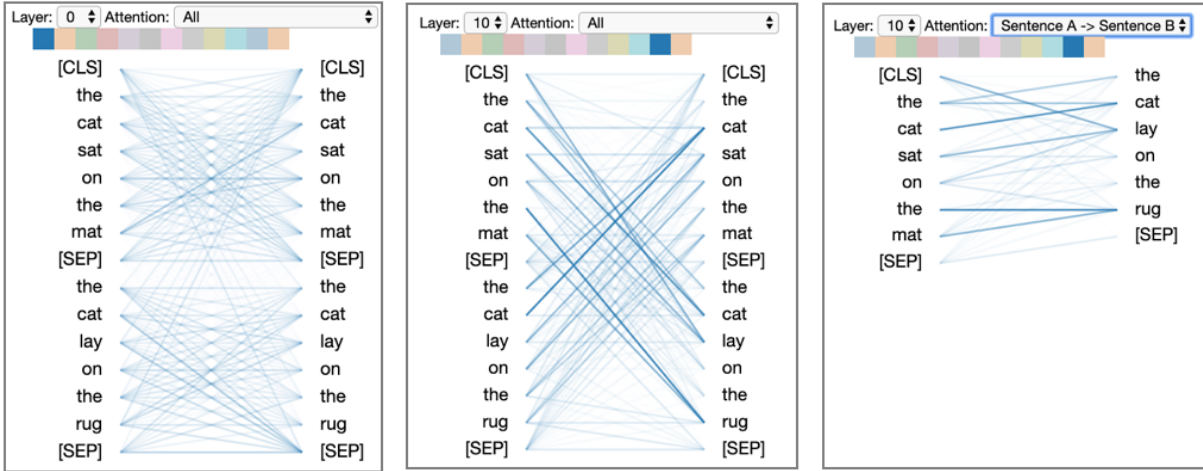

Figure 2: Attention head view for BERT, for inputs *the cat sat on the mat* (Sentence A) and *the cat lay on the rug* (Sentence B). The left and center figures represent different layers / attention heads. The right figure depicts the same layer/head as the center figure, but with *Sentence A → Sentence B* filter selected.

attended to (right). Colors identify the corresponding attention head(s), while line weight reflects the attention score.

At the top of the screen, the user can select the layer and one or more attention heads (represented by the colored patches). Users may also filter attention by token, as shown in Figure 1 (right); in this case the target tokens are also highlighted and shaded based on attention strength. For BERT, which uses an explicit sentence-pair model, users may specify a sentence-level attention filter; for example, in Figure 2 (right), the user has selected the *Sentence A → Sentence B* filter, which only shows attention from tokens in Sentence *A* to tokens in Sentence *B*.

Since the attention heads do not share parameters, each head can produce a distinct attention pattern. In the attention head shown in Figure 1 (left), for example, each word attends exclusively to the previous word in the sequence. The head

in Figure 1 (center), in contrast, generates attention that is dispersed fairly evenly across previous words in the sentence (excluding the first word). Figure 2 shows attention heads for BERT that capture sentence-pair patterns, including a within-sentence attention pattern (left) and a between-sentence pattern (center).

Besides these coarse positional patterns, attention heads may capture specific lexical patterns such as those shown in Figure 3. Other attention heads detected named entities (people, places, companies), beginning/ending punctuation (quotes, brackets, parentheses), subject-verb pairs, and a variety of other linguistic properties.

The attention-head view closely follows the original implementation from (Jones, 2017). The key difference is that the original tool was developed for encoder-decoder models, while the present tool has been adapted to the encoder-only BERT and decoder-only GPT-2 models.
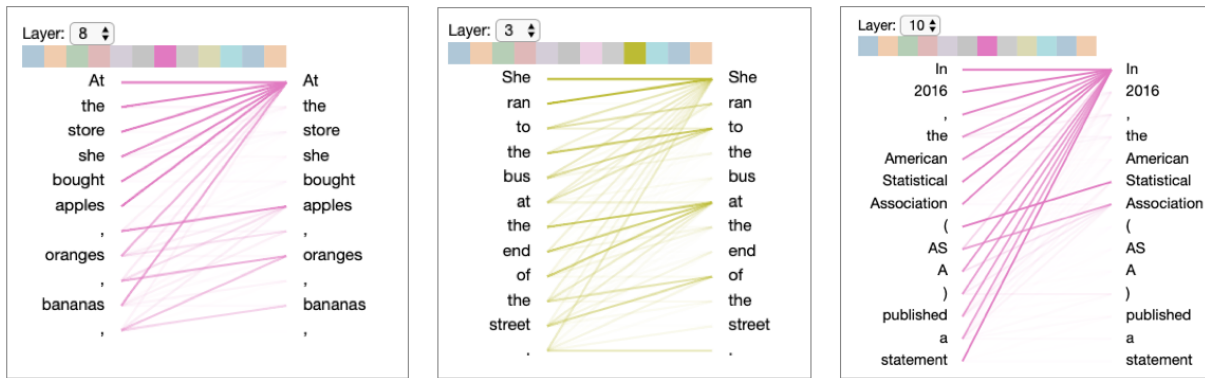
Figure 3: Examples of attention heads in GPT-2 that capture specific lexical patterns: list items (left); prepositions (center); and acronyms (right). Similar patterns were observed in these attention heads for other inputs. (Attention directed toward first token is likely null attention, as discussed later.)
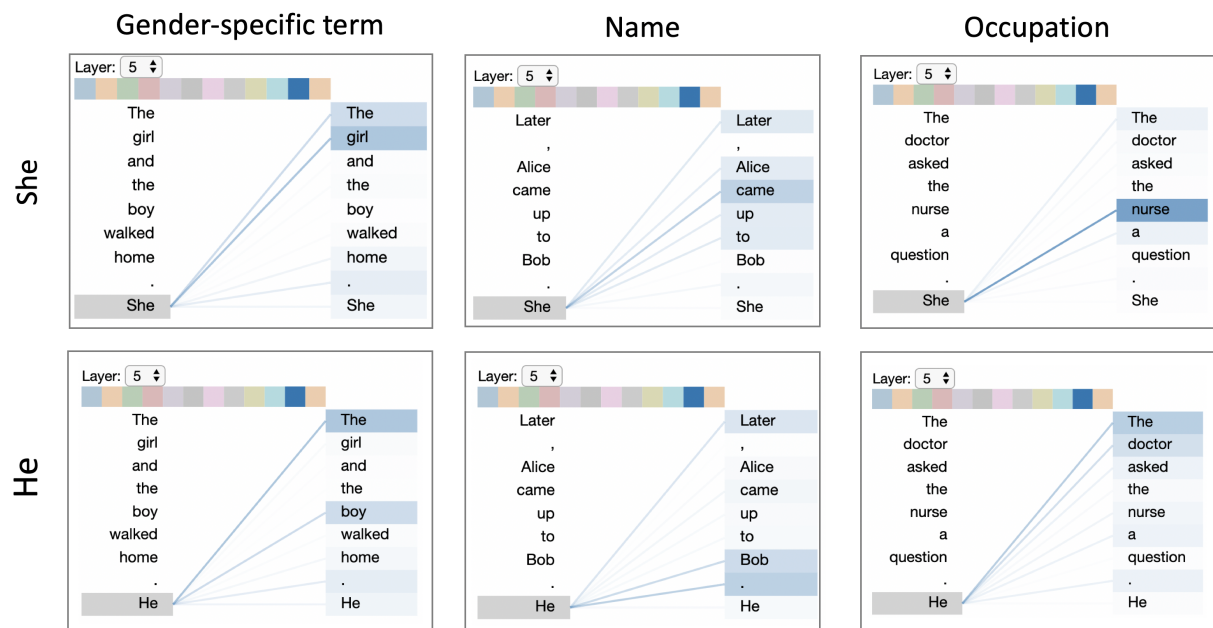

Figure 4: Attention pattern in GPT-2 related to coreference resolution suggests the model may encode gender bias.

Use Case: Detecting Model Bias

One use case for the attention-head view is detecting bias in the model. Consider the following two cases of conditional text generation using GPT-2 (generated text underlined), where the two input prompts differ only in the gender of the pronoun that begins the second sentence[4]:

- *The doctor asked the nurse a question. She said, "I'm not sure what you're talking about."*

- *The doctor asked the nurse a question. He asked her if she ever had a heart attack.*

In the first example, the model generates a continuation that implies *She* refers to *nurse*. In the

second example, the model generates text that implies *He* refers to *doctor*. This suggests that the model's coreference mechanism may encode gender bias (Zhao et al., 2018; Lu et al., 2018). To better understand the source of this bias, we can visualize the attention head that produces patterns resembling coreference resolution, shown in Figure 4. The two examples from above are shown in Figure 4 (right), which reveals that the token *She* strongly attends to *nurse*, while the token *He* attends more to *doctor*. This result suggests that the model is heavily influenced by its perception of gender associated with words.

By identifying a potential source of model bias, the tool can help to guide efforts to provide solutions to the issue. For example, if one were able

---

[4] Generated from GPT-2 small model, using greedy top-1 decoding algorithm

to identify the neurons that encoded gender in this attention head, one could potentially manipulate those neurons to control for the bias (Bau et al., 2019).

2.2 Model View

The *model view* (Figure 5) provides a birds-eye view of attention across all of the model's layers and heads for a particular input. Attention heads are presented in tabular form, with rows representing layers and columns representing heads. Each layer/head is visualized in a thumbnail form that conveys the coarse shape of the attention pattern, following the *small multiples* design pattern (Tufte, 1990). Users may also click on any head to enlarge it and see the tokens.
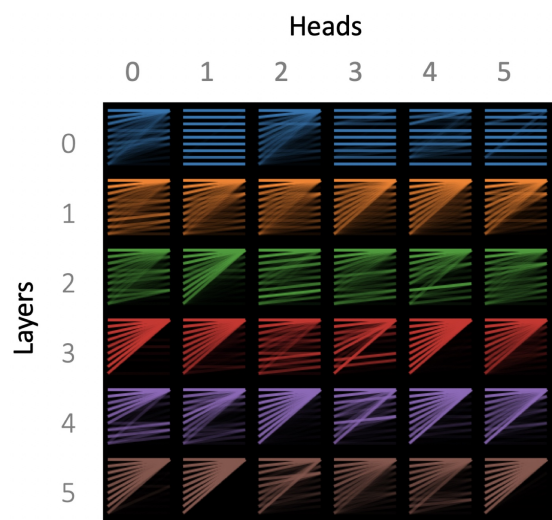

Figure 5: Model view of GPT-2, for input text *The quick, brown fox jumps over the lazy* (excludes layers 6-11 and heads 6-11).

The model view enables users to browse the attention heads across all layers in the model and see how attention patterns evolve throughout the model. For example, one can see that many attention heads in the initial layers tend to be position-based, e.g. focusing on the same token (layer 0, head 1) or focusing on the previous token (layer 2, head 2).

Use Case: Identifying Recurring Patterns
The model view in Figure 5 shows that many of the attention heads follow the same pattern: they focus all of the attention on the first token in the sequence. This appears to be a type of null pattern that is produced when the linguistic property captured by the attention head doesn't appear in the input text. One possible conclusion from this

result is that the model may benefit from a dedicated null position to receive this type of attention. While it's not clear that this change would improve model performance, it would make the model more interpretable by disentangling the null attention from attention related to the first token.

2.3 Neuron View

The *neuron view* (Figure 6) visualizes the individual neurons in the query and key vectors and shows how they are used to compute attention. Given a token selected by the user (left), this view traces the computation of attention from that token to the other tokens in the sequence (right). The computation is visualized from left to right with the following columns:

- **Query q**: The 64-element query vector of the token paying attention. Only the query vector of the selected token is used in the computations.

- **Key k**: The 64-element key vector of each token receiving attention.

- **q × k (element-wise)**: The element-wise product of the selected token's query vector and each key vector.

- **q · k**: The dot product of the selected token's query vector and each key vector.

- **Softmax**: The softmax of the scaled dot-product from previous column. This equals the attention received by the corresponding token.

Positive and negative values are colored blue and orange, respectively, with color saturation based on the magnitude of the value. As with the attention-head view, the connecting lines are weighted based on attention between the words. The element-wise product of the vectors is included to show how individual neurons contribute to the dot product and hence attention.

Use Case: Linking Neurons to Model Behavior
To see how the neuron view might provide actionable insights, consider the attention head in Figure 7. For this head, the attention (rightmost column) appears to decay with increasing distance from the source token[5]. This pattern resembles a context

---

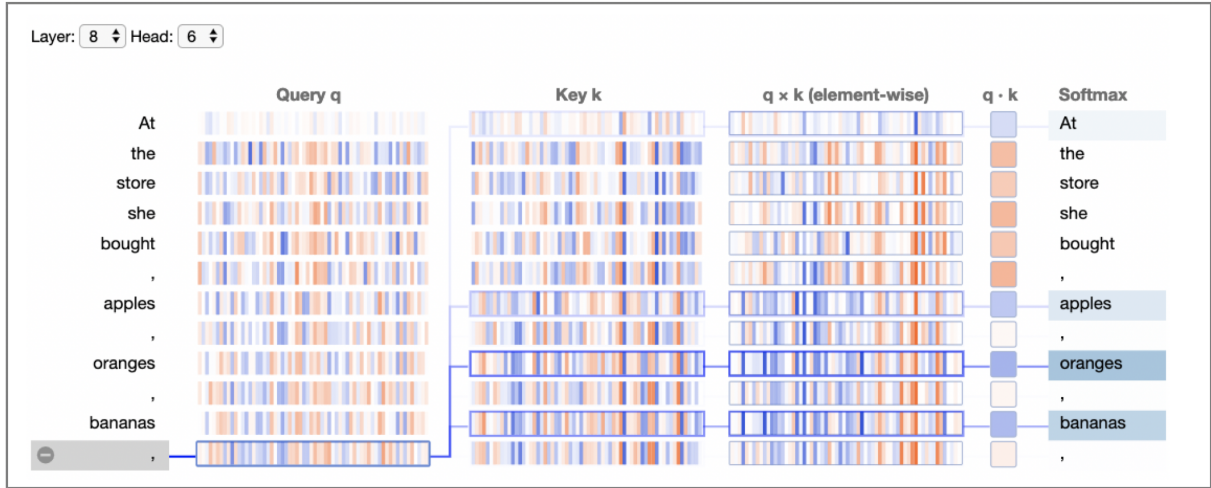[5] with the exception of the first token, which acts as a null token, as discussed earlier.

Figure 6: Neuron view of GPT-2 for layer 8 / head 6. Positive and negative values are colored blue and orange respectively, with color saturation reflecting magnitude. This is the same attention head depicted in Figure 3 (left).

window, but instead of having a fixed cutoff, the attention decays continuously with distance.

The neuron view provides two key insights about this attention head. First, the attention scores appear to be largely independent of the content of the input text, based on the fact that all the query vectors have very similar values (except for the first token). The second observation is that a small number of neuron positions (highlighted with blue arrows) appear to be mostly responsible for this distance-decaying attention pattern. At

these neuron positions, the element-wise product q×k decreases as the distance from the source token increases (either becoming darker orange or lighter blue).

When specific neurons are linked to a tangible outcome—in this the case decay rate of attention—it presents an opportunity for human intervention in the model. By altering the values of the relevant neurons (Bau et al., 2019), one could control the rate at which attention decays for this attention head. This capability might be use-
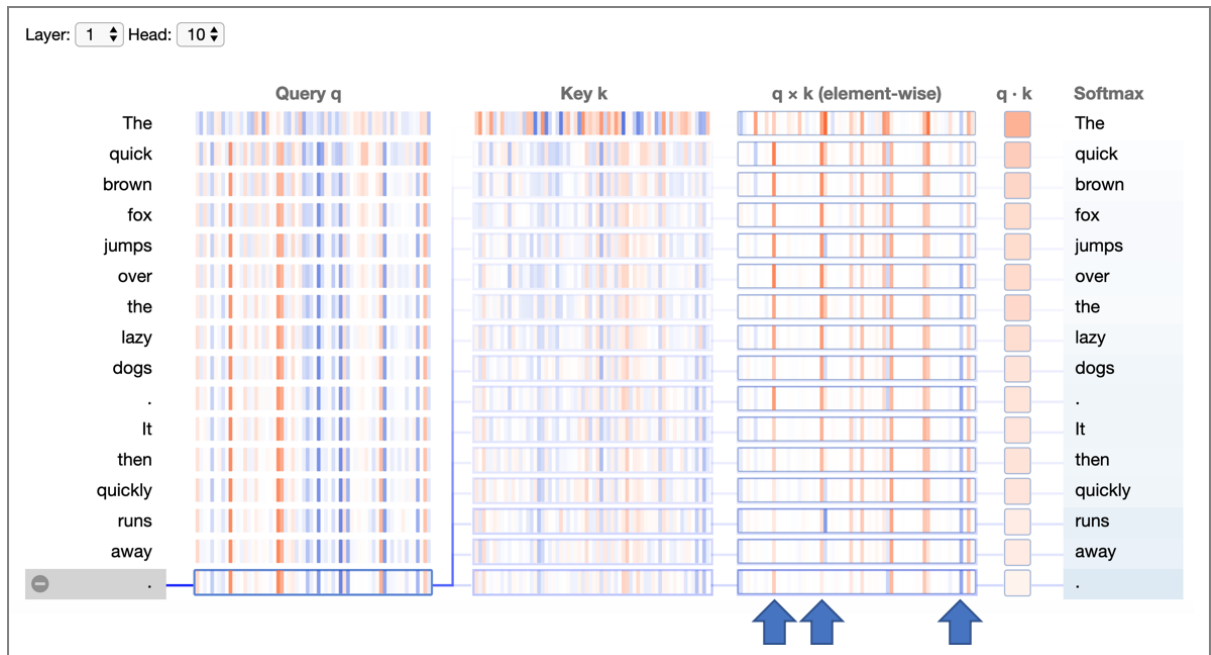

Figure 7: Neuron view of GPT-2 for layer 1 / head 10 (same one depicted in Figure 1, center) with last token selected. Blue arrows mark positions in the element-wise products where values decrease with increasing distance from the source token (becoming darker orange or lighter blue).

ful when processing or generating texts of varying complexity; for example, one might prefer a slower decay rate (longer context window) for a scientific text and a faster decay rate (shorter context window) for content intended for children.

3 Conclusion

In this paper, we presented a tool for visualizing attention in Transformer-based language representation models. We demonstrated the tool on the OpenAI GPT-2 and BERT models and presented three use cases for analyzing GPT-2. For future work, we would like to evaluate empirically how attention impacts model predictions across a range of tasks (Jain and Wallace, 2019). Further, we would like to integrate the three views into a unified interface, and visualize the value vectors in addition to the queries and keys. Finally, we would like to enable users to manipulate the model, either by modifying attention (Lee et al., 2017; Liu et al., 2018; Strobelt et al., 2018) or editing individual neurons (Bau et al., 2019).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Anthony Bau, Yonatan Belinkov, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2019. Identifying and controlling important neurons in neural machine translation. In *ICLR*.

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics (TACL) (to appear)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv Computation and Language*.

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

Llion Jones. 2017. Tensor2tensor transformer visualization. https://github.com/tensorflow/tensor2tensor/tree/master/tensor2tensor/visualization.

Jaesong Lee, Joong-Hwi Shin, and Jun-Seok Kim. 2017. Interactive visualization and manipulation of attention-based neural machine translation. In *EMNLP: System Demonstrations*.

Shusen Liu, Tao Li, Zhimin Li, Vivek Srikumar, Valerio Pascucci, and Peer-Timo Bremer. 2018. Visual interrogation of attention-based models for natural language inference and machine comprehension. In *EMNLP: System Demonstrations*.

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. 2018. Gender bias in neural natural language processing. *CoRR*, abs/1807.11714.

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

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomas Kocisky, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. In *ICLR*.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*.

H. Strobelt, S. Gehrmann, M. Behrisch, A. Perer, H. Pfister, and A. M. Rush. 2018. Seq2Seq-Vis: A Visual Debugging Tool for Sequence-to-Sequence Models. *ArXiv e-prints*.

Edward Tufte. 1990. *Envisioning Information*. Graphics Press, Cheshire, CT, USA.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N. Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. 2018. Tensor2tensor for neural machine translation. *CoRR*, abs/1803.07416.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017a. Attention is all you need. In *Advances in Neural Information Processing Systems*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017b. Attention is all you need. Technical report.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.