Author manuscript, published in Proceedings of the IEEE VIS Conference, pp. 146-150, 2019.
DOI: 10.1109/VISUAL.2019.8933677

SANVis: Visual Analytics for Understanding Self-Attention Networks

Cheonbok Park* — Korea University
Inyoup Na* — Korea University
Yongjang Jo† — Korea University
Sungbok Shin‡ — University of Maryland
Jaehyo Yoo* — Korea University
Bum Chul Kwon§ — IBM Research
Jian Zhao¶ — University of Waterloo
Hyungjong Noh‖ — NCSOFT Co., LTD.
Yeonsoo Lee‖ — NCSOFT Co., LTD.
Jaegul Choo* — Korea University

Figure 1: Overview of SANVis. (A) The control panel presents three types of visualization options: (A-1) the attention piling view, (A-2) the Sankey view, and (A-3) the small multiples view. (B) The network view displays multiple attentions for each layer according to a selected visualization option. (B-2) Different color bar heights indicate the average attention weights based on different heads (eight heads in total) at each layer. (C) The HeadLens helps the user analyze what the attention head learned by showing representative words and by providing statistical information of part-of-speech tags and positions.

Abstract

Attention networks, various deep neural network architectures inspired by humans' attention mechanism, have seen significant success in image captioning, machine translation, and many other applications. Recently, they have been further evolved into highly complicated structures that simultaneously use multiple attentions, called multi-head attentions, to achieve state-of-the-art performances. Despite the outstanding performances, the complexity prevents users from easily understanding and manipulating the inner workings of models. To tackle the challenges, we present a visual analytics system called SANVis, which helps users understand the behaviors and the characteristics of attention modules of a particular layer as well as those which contain multi-head attention modules. Using a state-of-the-art self-attention model called Transformer, we demonstrate how the design of SANVis can be useful to visually explore the inner workings of the model for machine translation tasks.[1]

Index Terms: Deep learning—natural language processing—self-attention networks—model interpretation

*{cb_park, windy9898, vkfwlsdhs, jchoo}@korea.ac.kr
†jyj3312@gmail.com
‡sbshin90@cs.umd.edu
§bumchul.kwon@us.ibm.com
¶jianzhao@uwaterloo.ca; work was completed while at FXPAL.
‖{nohhj0209,yeonsoo}@ncsoft.com

[1]Our system is available at `http://short.sanvis.org`.

1 Introduction

Attention-based deep neural networks, inspired by humans' attention mechanism, are widely used for sequence-to-sequence modeling, e.g., machine translation of a sentence (a sequence of words) in one language to that in another. The attention module allows the model to dynamically utilize different parts of the input sequence, which leads to state-of-the-art performances in natural language processing (NLP) tasks [4, 13, 32].

Recently, Vaswani et al. [26] proposed advanced, multi-head self-attention networks called Transformer, which captures diverse syntactic and semantic information across a sequence of words in a given text. Transformer has significantly improved state-of-the-art performances of machine translation, compared with conventional approaches using recurrent neural networks (RNNs). This model have been successfully applied to other NLP tasks [7, 19], as well as even computer vision ones [31, 35].

The success of self-attention stems from its parallel, multi-headed architecture. Multi-head self-attention networks possess the following advantages: (1) They can properly model long-range dependencies among words in a sequence unlike RNN-based models that have a limited capability in this respect. (2) Furthermore, they can simultaneously capture different types of syntactic and semantic relationships among words, via different attention heads of which each projects word vectors into different latent subspaces. Simultaneously utilizing such differently projected information enhances the performance of the model for various NLP tasks.

However, the recent advancement in attention networks brings new challenges. The highly sophisticated network structure prevents users from understanding computational processes and using the models for various analytic tasks. In NLP domains, recent stud-

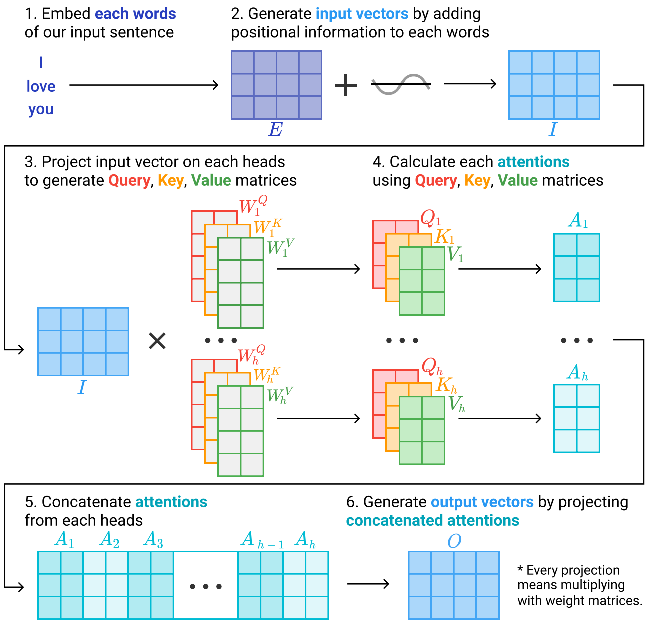

Figure 2: How a multi-head self-attention module works. Steps 1 and 2 correspond to the embedding layer, while Steps 3 to 6 correspond to a single-layer multi-head self-attention example.

ies [6, 7, 27] aim to analyze the inner-workings of self-attention models. Such analysis helps users improve the model, such as in removing unnecessary heads and refining them. Consequently, recent studies [24, 28] attempt to understand questions such as what kinds of features the model learned differently in heads or which head captured a specific set of linguistic features. To the best of our knowledge, our work is one of the first visualizations that is designed to help users understand the inner-workings of self-attention models.

This paper presents a visual analytics system called SANVis, which supports the user's understanding and interactive exploration of multi-head, self-attention networks. The contributions of our work are as follows: First, we introduce a novel visual analytics system called SANVis that helps users decipher models trained with advanced multi-head self-attention networks. Second, the usage scenario demonstrates that the harmonious integration of various views, interactive features, and Transformer is useful for users to gain valuable insights.

2 Related Work

Visual analytics approaches for various deep neural network architectures in diverse problem domains have been actively studied. There exist various visual analytics approaches for convolutional neural networks mainly in computer vision domains [2, 11, 12, 18, 25, 34] as well as those for RNNs mainly in NLP domains [5, 10, 16, 22, 23].

Other advanced types of deep neural networks have been integrated into a visual analytics framework, such as generative adversarial networks [8,30], deep reinforcement learning [29]. In an attention model case, Strobelt et al. [21] propose a new technique to visualize the RNN-based attention model. It helps to explore and understand the components of a sequence to sequence model. By showing each step in the inner process of the model, it supports to interpret the complex mechanism of the model. These systems allow users to understand the complicated inner mechanism of the advanced deep learning model. However, to our knowledge, despite the success of BERT [7] and Transformer, visual analytics approaches for advanced attention networks involving multi-head self-attention have not existed before, so ours is the first visual analytics system for multi-head self-attention networks.

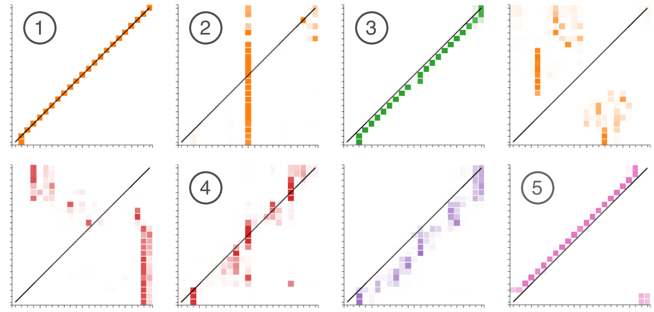

Figure 3: Diverse attention pattern examples in the encoder of the Transformer. Some attention heads show diagonal patterns indicating that a query word attends to itself (1) or its immediate previous (5) or next word (3). Some other attention heads attend to a single word (2). In other attention heads, close query words attend to same words (4).

3 Background of Self-Attention Networks

In this section, we focus on briefly reviewing the attention module, called Transformer [26]. Transformer adopts an encoder-decoder architecture to solve sequence-to-sequence learning tasks. Transformer turns a sequence of words in one domain into that in another domain. For example, for machine translation tasks, it translates a sentence in one language into that in another language. In this process, the encoder of Transformer converts input words (e.g., English words) to internal, hidden-state vectors, and the decoder turns the vectors into a sequence of output words (e.g., French words).

Each encoder and decoder respectively consists of multiple layers of computing functions inside. Furthermore, each layer in the encoder includes two sequential sub-layers, which are a multi-head self-attention and a position-wise feed-forward network. In addition to the multi-layer architecture of the encoder, the decoder has an additional attention layer, which called as an encoder-decoder attention and helps the model to give attention to the encoders' internal states. Each layer of both encoder and decoder also consists of skip-connection and layer normalization in their computation pipeline. Overall encoder and decoder architecture are the stacks of L identical encoder layers or decoder layers, including an embedding layer.

We summarize the computation process with mathematical notations, so readers are advised to read the remaining section for details: Let us denote d_{model} as the size of hidden(internal) state vector and h as the number of heads in multi-head self-attention. Each dimension of query, key, and value vector is $d_q = d_k = d_v = d_{model}/h$.

The embedding layer transforms the input token x_i to its embedding space e_i using a word embedding and adds the position information for each input token using sinusoidal functions (see Steps 1 and 2 in Fig. 2), where x_i is the i-th input token in $X = [x_1, \cdots, x_T]$.

At each attention head, we transform encoded word vectors into three matrices of a query, a key, and a value, $Q \in R^{T \times d_q}$, $K \in R^{T \times d_k}$, and $V \in R^{T \times d_v}$, respectively, for h times, which in turn generated $h \times 3$ matrices, using the linear transformation and compute the attention-weighted combinations of value vectors as

$$\begin{aligned} \text{Attention}(Q,K,V) &= \text{Softmax}\left(\frac{QK^T}{\sqrt{d_{model}}}\right)V \\ \text{MultiHeadAttention} &= \text{Concat}\left(head_1, \ldots, head_h\right)W^O \end{aligned} \tag{1}$$

where $head_i = \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right)$, and W_i^Q, W_i^V and W_i^K indicate the linear transformation matrices at the i-th head. In multi-head self-attention, which consists of h parallel attention heads, transformation matrices of each head are randomly initialized, and then each set is used to project input vectors onto a different representation subspace. For this reason, every attention head is allowed to have different attention shapes and patterns. This characteristic encourages each head differently to attend adjacent words or linguistics relation words.

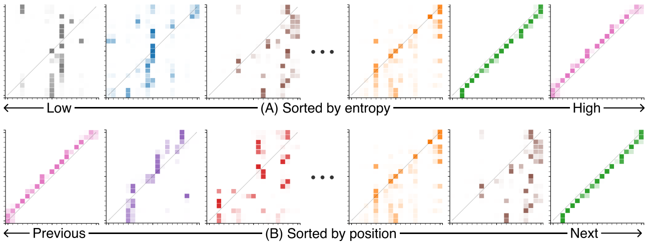

Figure 4: Attention sorting result. The user can sort a set of multiple attention patterns with respect to different criteria such as the entropy measure (A) and the relative positional offset from query words (B).

In the encoder layer, source words (input words to the encoder) work as the input to the query, key, and value transformations at the i-th head. In the decoder layer, the input can vary by attention types. While the decoders' self-attention takes target words (output words of the decoder) as its input, the encoder-decoder attention has target words as input to a query transformation but source words as the input to a key and a value transformation.

4 Goals and Tasks

SANVis helps researchers understand and effectively analyze numerous attention heads in self-attention. The goal can be broken down into three user tasks:

Task T1: Gain an overview of self-attention models. The user understands the information flow along the layer.

Task T2: Detect and compare patterns from multiple attention heads. The user quickly explores the attention patterns and find distinct patterns by comparing with attention from other heads.

Task T3: Understand the characteristics of the inner-working mechanism. The user investigates whether the model captures the positional or linguistic characteristics.

5 SANVis

We present SANVis, a visual analytics system for the in-depth understanding of the self-attention models, as shown in Fig. 1. SANVis provides various visualization modules at different views: (1) network overview allows the user to understand the overall information flow through our visualization across the multiple layers (T1), (2) a single layer views that visualizes attention patterns of multiple heads within a layer (T2), and (3) a HeadLens that reveals the characteristics of the query and the key vectors and their relationship of a particular head (T3).

5.1 Network Overview

SANVis mainly visualizes the overview of attention propagation patterns across multiple layers using the Sankey diagram (T1). As shown in Fig. 1 (B), a set of words are aligned vertically in each layer, and the edge weight between them represents the average attention weight among multiple heads within a particular layer. In Fig. 1 (B-1), one can see the strong link that stretches from 'physically' in layer 2 to that in layer 3. It means a significant amount of information of 'physically' in layer 2 is conveyed to encode that word in layer 3.

SANVis shows the histogram of each word in Fig. 1 (B-2). Each bar corresponds to each head within the layer where its height represents the total amount of attention weights assigned to those words by a specific head. As with Fig 1 (B-2), if the fourth head in the layer attended to the word 'planet' more highly than others, the fourth bar would be higher than the others. In this manner, SANVis shows not only the overall attention flow but also those influential words assigned high attention weights by a particular head. Additionally, when the user moved the mouse over the fourth color bar, we show an attention heat map of the fourth head that layer.

We provide an additional control panel to interact with this multi-layer-level view (Fig. 1 (A)). For example, one can replace the current Sankey diagram with a heatmap view, where multiple heatmaps corresponding to different heads can be sorted by various criteria. Additionally, SANVis also provides the attention piling option to aggregate multiple attention patterns into a small number of clusters.

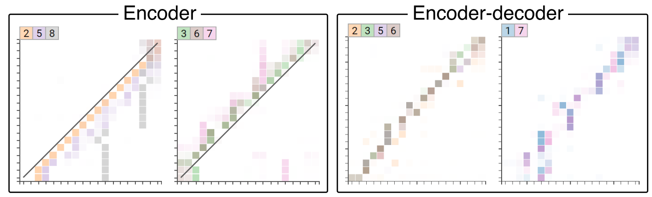

Figure 5: Attention piling example in the encoder layer and encoder-decoder layer. In the encoder-decoder example, piling results do not have a diagonal line because of the difference between the count of query words and key words.

5.2 Single Layer Views (Involving Multi-Heads)

Unlike the traditional RNN-based attention models that contain only one attention head in the entire model, recent models involve multiple attention heads in a single layer, and even worse, the number of heads tends to increase in these days. As a result, it is challenging to grasp the patterns of multiple different attentions simultaneously. To address this issue, SANVis provides 'piling' and 'sorting' capabilities to understand common as well as distinct attention patterns among multiple attention heads (T2).

Attention Sorting. Fig. 3 shows various attention patterns between query (y-axis) and key (x-axis) words for different attention head in different layers. We focused on reducing the users' efforts, which is to find the distinguish attention patterns, based on relative positional information and entropy (Fig. 3). Relative positional information, such as whether the attention goes mainly toward the left, right, or the current location, as well as the column-wise mean entropy value of the attention matrix, were obtained to allow the users to detect these patterns easily.

Fig. 4 shows the sorting results of attentions based on our position or entropy sorting algorithms. When sorted by position, a number of attention was unambiguous that attention that inclines towards the past words were placed near the control panel at the top while those that lean towards the future words were placed relatively close to the bottom. When sorted by entropy, the uppermost attention had the lowest entropy and exhibited bar-shaped attention, which numerous query words attend the same word. At the bottom, the user can find that no more words focused on the same word.

Attention Piling. Inspired by the heatmap piling methods [3, 20], we applied this piling idea to summarize multiple attention patterns in a single layer, as shown in the encoder part of Fig. 5 . To this end, we compute the feature vector of each attention head and perform clustering to form piles (or clusters) of attention.

The feature vector of a particular attention on attention head is defined as a flattened n^2-dimensional vector of its $A_i \in R^{T \times T}$ attention matrix, where A_i is calculated from Softmax $\left(\frac{QK^T}{\sqrt{d_{model}}}\right)$ on the i-th head, concatenated with additional three-dimensional vector of (1) the sum of the upper triangular part of the matrix, (2) that of the lower-triangular part, and (3) the sum of diagonal entries. This three-dimensional vector indicates the proportions how much attention is assigned to (1) the previous words of a query word, (2) its next words, (3) and the query itself, respectively.

Using these feature vectors of multiple attention heads within a single layer, SANVis performs hierarchical clustering based on their Euclidean distances. In this manner, multiple attention patterns are grouped, forming an aggregated heatmap visualization per computed pile along with head indices belonging to each pile, as shown in Fig. 5. It helps the user easily find the similar patterns and distinct patterns in the same layer by adjusting Euclidean distance.

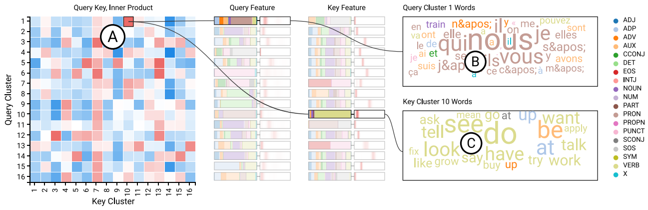

Figure 6: HeadLens example in the encoder-decoder attention of head 7 in layer 4.

5.3 HeadLens

To analyze a particular attention head, SANVis offers a novel view called the HeadLens, as shown in Fig. 1 (C). This view facilitates detailed analysis of the query and key representations of the selected attention head, such as which linguistic or positional feature they encoded and captured. (T3) This view gets open when a user clicks a particular heatmap corresponding to an attention head in the network overview panel.

This view works in the following steps: (1) A set of clusters of query vectors as well as those of key vectors are first obtained. (2) We can obtain the centroid vectors for each cluster of query and key vectors. (3) All their pairwise similarities between the query centroid and the key centroid are computed, as shown as a heatmap in Fig. 1 (C-2). (4) Additionally, the POS tagging and the positional information is summarized for each of the query and the key clusters (Fig. 1 (C-3)). (5) Once a user clicks a particular cell with a high (or low) similarity value with a red (or blue) color, its corresponding query and a key cluster are summarized in terms of their representative keywords (Fig. 1 (C-4)).

To be specific, in the first step, we consider all the sentences in a validation set and obtain the query and the key vectors of all the words from these sentences. These query and key vectors are the results of applying a query and a key transformation of input words for a given attention head. Next, we perform the K-means++ [17] algorithm for each of the above-described query and key vector sets, by using the pre-defined number of clusters, e.g., 16 in our case. We empirically set this number of clusters by using an elbow method.

In the second step, we obtain the cluster centroid vectors from the set of clusters for query vectors as well as those centroid vectors for key vectors. In the third step, we compute all the pairwise inner product similarities between each pair of a query cluster centroid and a key cluster one, which are visualized as a heatmap (Fig. 1 (C-2)). We chose the inner product as a similarity measure since the attention weight is mainly computed based on the inner product between a query and a key vector. In practically, high inner product between a query set and a key set means that the words in the query set are likely to attend to the words in the key set.

In the third step, the HeadLens provides a summary of each of the query and the key clusters. Each query (or key) cluster contains those words whose query (or key) vectors belong to the cluster. For those words, we obtain their part-of-speech (POS) tags and position indices within the sentence which each of them appears in. For POS tags, we used universal POS tagger [14]. Afterward, the relative amount of those words with each POS tag type out of the entire words within a single cluster is shown as a horizontal bar width with its encoded color, as shown in the left bar of Fig. 1 (C-3). In addition, the relative amount of those words shown in a particular position of their original sentences are color-coded (a higher value colored as a red), as shown in the right bar of Fig. 1 (C-3).

Finally, in the fourth step, the user can click a particular entry in the cluster-level heatmap, e.g., a pair of a query and a key cluster with high similarity (a red cell highlighted in a black square in Fig. 1 (C-2)). Then, the summary of the corresponding query and key cluster are indicated by a black-colored edge (Fig. 1 (C-3)). Additionally, the word cloud visualization of such user-selected query and key cluster are used to highlight the frequently appearing words in each cluster, color-coded with their own POS tag types (Fig. 1 (C-4)).

For example, the selected entry in Fig. 1 (C-2) indicates that the query cluster 15 has high similarity with the key cluster 15. The selected query cluster mainly contains auxiliary verb words (orange-colored), while the selected key cluster mainly contains noun words (purple-colored) in Fig. 1 (C-3). Furthermore, their most appearing words are shown in the word cloud view (Fig. 1 (C-3)), which means that this head assigns a high attention weight to these noun words, e.g., 'world' and 'life' when a query word is given as an auxiliary verb word, e.g., 'is' and 'are.' This result shows that the selected head have captured the linguistic relationship of noun and verb.

6 Usage Scenarios

This section demonstrates usage scenarios of SANVis, mainly focusing on the recently proposed Transformer. This model has shown superior performances in machine translation tasks, including English-French and English-German translation tasks in the WMT challenger [1]. Our implementation of the Transformer is based on the annotated Transformer [9]. Our model parameter setting followed the base model in the original paper [26]. We set our target task as English-French translation, where the collection of the scripts from TED lectures is used as our dataset [15]. The BLEU score of our model is shown as 38.4, which validates a reasonable level of performance. For evaluating our system, we used the validation set, which is not seen during training.

Attention Piling. In Fig. 5, the encoder-decoder part shows the attention piling visualization in encoder-decoder attention. In this example, one can observe that a number of attention heads have a diagonal attention pattern. An appropriate explanation of this diagonal shape would be that the words in French and English are generally aligned in the same order [4]. For the debugging purpose, it proves that the model properly learned a linguistic alignment between the source sequence and the corresponding target sequence.

HeadLens. In the earlier example, we saw that most attention patterns between the English and the French words are diagonally-shaped between English and French words. One can analyze this pattern in detail by using our HeadLens. As shown in Fig. 6, we chose head 4 in layer 7, which has such a diagonal attention pattern, and applied the HeadLens. Once selecting the query and the key cluster pair with a high similarity (Fig. 6 (A)), it is shown that the query clusters commonly have an pronoun as a dominant POS tag type (brown-colored in Fig. 6 (B)). Most of query cluster words are subject words in French, for instance, 'nous' and 'vous' mean 'we' and 'you' in English, respectively. The corresponding key clusters' representative words are mostly verbs. This result demonstrates that the model attends verb words to predict verb tokens for translating from English to French, when the input token is subject.

7 Conclusions

In this paper, we present SANVis, a visual analytics system for the self-attention networks that supports in-depth understanding of multi-head self-attention networks at various levels of granularity, such as a multi-layer and a single layer. In the usage scenario, we demonstrate that our system provides the user with a deep understanding of the multi-head self-attention model in machine translation.

As future work, we plan to extend our HeadLens to perform clustering of value vectors. We evaluate our system by various researchers who use the multi-head self-attention networks. We also apply our method in other state-of-the-art self-attention based models, such as BERT [7] and XLNet [33].

Acknowledgments

The authors wish to thank all reviewers who provided constructive feedback for our project. This work was partially supported by NCSOFT NLP Center.

REFERENCES

[1] ACL 2014 Ninth Workshop on Statistical Machine Translation.

[2] B. Alsallakh, A. Jourabloo, M. Ye, X. Liu, and L. Ren. Do convolutional neural networks learn class hierarchy? *CoRR*, abs/1710.06501, 2017.

[3] B. Bach, N. H. Riche, T. Dwyer, T. M. Madhyastha, J.-D. Fekete, and T. J. Grabowski. Small multipiles: Piling time to explore temporal patterns in dynamic networks. *Comput. Graph. Forum*, 34:31–40, 2015.

[4] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[5] D. Cashman, G. Patterson, A. Mosca, N. Watts, S. Robinson, and R. Chang. RNNbow: Visualizing Learning Via Backpropagation Gradients in RNNs. *IEEE Computer Graphics and Applications*, 38(6):39–50, Nov. 2018.

[6] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning. What does bert look at? an analysis of bert's attention. *ArXiv*, abs/1906.04341, 2019.

[7] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[8] M. Kahng, N. Thorat, D. H. Chau, F. Viégas, and M. Wattenberg. GAN Lab: Understanding Complex Deep Generative Models using Interactive Visual Experimentation. *ArXiv e-prints*, Sept. 2018.

[9] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush. Opennmt: Open-source toolkit for neural machine translation. In *Proc. ACL*, 2017.

[10] B. C. Kwon, M. Choi, J. T. Kim, E. Choi, Y. B. Kim, S. Kwon, J. Sun, and J. Choo. RetainVis: Visual Analytics with Interpretable and Interactive Recurrent Neural Networks on Electronic Medical Records. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):299–309, Jan. 2019.

[11] D. Liu, W. Cui, K. Jin, Y. Guo, and H. Qu. DeepTracker: Visualizing the Training Process of Convolutional Neural Networks. *ArXiv e-prints*, Aug. 2018.

[12] M. Liu, J. Shi, Y. Li, C. Li, J. Zhu, and S. Liu. Towards better analysis of deep convolutional neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 23:91–100, 2016.

[13] T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412–1421. Association for Computational Linguistics, Lisbon, Portugal, Sept. 2015.

[14] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55–60. Association for Computational Linguistics, Baltimore, Maryland, June 2014.

[15] P. Michel and G. Neubig. Extreme adaptation for personalized neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 312–318. Association for Computational Linguistics, 2018.

[16] Y. Ming, S. Cao, R. Zhang, Z. Li, Y. Chen, Y. Song, and H. Qu. Understanding hidden memories of recurrent neural networks. *CoRR*, abs/1710.10777, 2017.

[17] R. Ostrovsky, Y. Rabani, L. J. Schulman, and C. Swamy. The effectiveness of lloyd-type methods for the k-means problem. *J. ACM*, 59(6):28:1–28:22, Jan. 2013.

[18] N. Pezzotti, T. Höllt, J. V. Gemert, B. P. F. Lelieveldt, E. Eisemann, and A. Vilanova. Deepeyes: Progressive visual analytics for designing deep neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):98–108, Jan 2018.

[19] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019.

[20] N. H. Riche, J.-D. Fekete, and M. J. McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13, 2007.

[21] H. Strobelt, S. Gehrmann, M. Behrisch, A. Perer, H. Pfister, and A. M. Rush. Seq2seq-vis: A visual debugging tool for sequence-to-sequence models. *CoRR*, abs/1804.09299, 2018.

[22] H. Strobelt, S. Gehrmann, M. Behrisch, A. Perer, H. Pfister, and A. M. Rush. SEQ2seq-VIS : A Visual Debugging Tool for Sequence-to-Sequence Models. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):353–363, 2019.

[23] H. Strobelt, S. Gehrmann, H. Pfister, and A. M. Rush. Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):667–676, Jan 2018.

[24] E. Strubell, P. Verga, D. Andor, D. Weiss, and A. McCallum. Linguistically-informed self-attention for semantic role labeling. *ArXiv*, abs/1804.08199, 2018.

[25] C. Sunghyo, S. Sangho, P. Cheonbok, K. Kyeongpil, C. Jaegul, and K. Bum Chul. Revacnn: Steering convolutional neural network via real-time visual analytics. *FILM at NIPS - Future of Interactive Learning Machines Workshop*, 2016.

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. Curran Associates, Inc., 2017.

[27] J. Vig and Y. Belinkov. Analyzing the structure of attention in a transformer language model. *ArXiv*, abs/1906.04284, 2019.

[28] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *ArXiv*, abs/1905.09418, 2019.

[29] J. Wang, L. Gou, H. Shen, and H. Yang. Dqnviz: A visual analytics approach to understand deep q-networks. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):288–298, Jan 2019.

[30] J. Wang, L. Gou, H. Yang, and H.-W. Shen. Ganviz: A visual analytics approach to understand the adversarial game. *IEEE Transactions on Visualization and Computer Graphics*, 24:1905–1917, 2018.

[31] X. Wang, R. B. Girshick, A. Gupta, and K. He. Non-local neural networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7794–7803, 2018.

[32] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In F. Bach and D. Blei, eds., *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37 of *Proceedings of Machine Learning Research*, pp. 2048–2057. PMLR, Lille, France, 07–09 Jul 2015.

[33] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. *ArXiv*, abs/1906.08237, 2019.

[34] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Computer vision–ECCV 2014*, pp. 818–833. Springer, 2014.

[35] H. Zhang, I. J. Goodfellow, D. N. Metaxas, and A. Odena. Self-attention generative adversarial networks. *CoRR*, abs/1805.08318, 2018.