Visual Interrogation of Attention-Based Models for Natural Language Inference and Machine Comprehension

Shusen Liu[1], Tao Li[2], Zhimin Li[3], Vivek Srikumar[2], Valerio Pascucci[3], Peer-Timo Bremer[1]
Lawrence Livermore National Laboratory[1]
School of Computing, University of Utah[2]
SCI Institute, University of Utah[3]

Abstract

Neural networks models have gained unprecedented popularity in natural language processing due to their state-of-the-art performance and the flexible end-to-end training scheme. Despite their advantages, the lack of interpretability hinders the deployment and refinement of the models. In this work, we present a flexible visualization library for creating customized visual analytic environments, in which the user can investigate and interrogate the relationships among the input, the model internals (i.e., attention), and the output predictions, which in turn shed light on the model decision-making process.

1 Introduction

Deep neural networks have been successfully applied to various natural language processing tasks. As the design of neural networks evolves, there is a clear trend of increasing model complexity in terms of both the architecture and the parameter count. Although expressive models help improve the prediction performance, the fundamental lack of interpretability leads many researchers to consider neural network models black boxes. The ability to interpret model internals and reason about predictions is essential for understanding the limitation of the models and improving upon them.

Recently, the attention networks have recently become widely-adopted (Bahdanau et al., 2015; Seo et al., 2017; Parikh et al., 2016b; Vaswani et al., 2017). Attention not only improves the model performance but also yields interpretable intermediate representations (i.e., alignment among words). As a result, attention can provide a natural interface for analyzing the internals of neural networks. Conducting analysis directly on the raw attention values can be challenging for human users, and therefore, visual representations that highlight word alignments between sentences have been proposed, such as the bipartite graph and the heatmap of the attention matrix (Li et al., 2015, 2016; Lee et al., 2017). However, many challenges remain. Firstly, for some NLP tasks, the word sequences pair for which the attention is computed can be highly asymmetrical (i.e., in machine comprehension, the context paragraph can be much longer than the question sentence), which the standard visual encoding cannot adequately handle. Also, many previous works present the attention in a static setting. However, the ability to provide an interactive environment in which the user can instantaneously look at how changes in the input affect the attention, and how small variations in the attention alter the prediction, is crucial for interpreting the model. Finally, many previous visualization efforts, despite revealing many exciting results, focus more on illustrating potentially useful visualization techniques than on providing a flexible software tool that can be easily integrated into an existing code base. As a result, applying these proposed techniques to real-world examples may involve substantial engineering effort, which could prevent wide adoption.

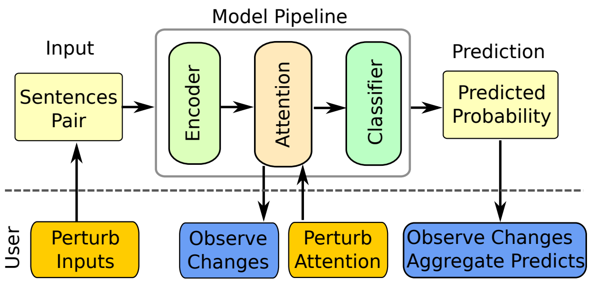


Figure 1: Perturbation-driven interrogation of the end-to-end models that follow the encoder, attention, and classifier structure. The user can generate a small perturbation of the input (i.e., replacing synonyms, paraphrasing), edit the attention, and observe the changes.

To address these challenges in interpreting attention-based models, we introduce a Python library that allows users to create web-based interactive visual analytics environments, in which they can interrogate the model by perturbing the input text and observing the effects on the attention and prediction, or modifying the attention and studying how it affects predictions (see Figure 1). We demonstrate our library applied to two NLP tasks: natural language inference (NLI) and machine comprehension (MC).

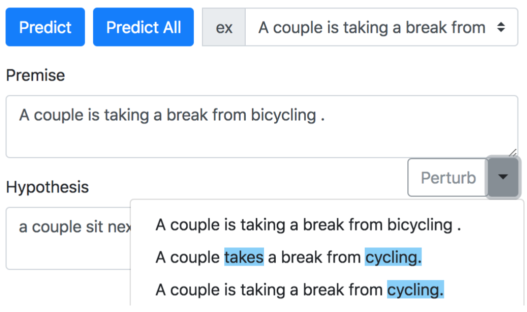
2 Related Works

Due to the increasing demand for model interpretability, many previous works have been proposed for making sense of NLP models by examining individual predictions as well as the model mechanism as a whole. In recent work, Li et al. (2015) investigated the composability of the vector-based text representations using instance-level attribution techniques that originated from the vision community (e.g., Zeiler and Fergus, 2014). In a study of the representation of erasure, Li et al. (2016) explained neural model decisions by exploring the impact of altering or removing the components of the model (i.e., changing the dimension count of hidden units or input words) on the prediction performance.

Besides interpreting the model via carefully designed experiments, several interactive demo/visualization systems, such as AllenNLP's demos (`http://demo.allennlp.org/`), often rely on visual encodings to summarize the model predictions. These systems provide a flexible environment in which the user can experiment with the various inputs and perform error analysis. The hidden state properties of the LSTM are visualized and investigated in the LSTMvis visualization system (Strobelt et al., 2018). Lee et al. (2017) visualized the beam search and attention component in neural machine translation models, in which the user can dynamically change the probability for the next step of the search tree or change the weight of the attention. In the visualization work on question answering (Rücklé and Gurevych, 2017), the system shows the text context and highlights the critical phrase that is used to answer the question.

3 Visualization System

As illustrated in Figure 1, many recent end-to-end NLP models follow a similar encoder–attention–classifier structure. The attention stage provides a window to peek into the model decision-making process. However, the static attention alone does not tell the whole story. Our proposed system uses a perturbation-driven exploration strategy that allows the user to manually or automatically perturb part of the pipeline and observe the changes downstream. The system consists of several visualization components that can be selectively combined to form a functional interactive system for a given task. In the following section, we will cover some of the essential components; the usage of all components can be found in the accompanying video link.



Premise

A couple is taking a break from bicycling .

Hypothesis

a couple sit nex

A couple is taking a break from bicycling .

A couple takes a break from cycling.

A couple is taking a break from cycling.

Figure 2: The interface for showing input sentences. The user can manually edit the words or apply automatic perturbation/paraphrasing of the inputs. In the "perturbed" drop-down menu, the blue background highlights words not in the original sentence.

3.1 Input Sentences Perturbation

Due to the discrete nature of natural language, automatically perturbing a sentence for sensitivity analysis can be particularly challenging (compared to other domains such as images) — small changes in words can lead to drastic differences in the semantics of the sentences. To reduce the potential semantic deviation, we allow for a straightforward perturbation method by replacing *nouns* and *verbs* by their synonyms in WordNet (Miller, 1995). However, synonym replacement does not guarantee that the meaning of the sentence remains the same. Furthermore, WordNet often produces rare words or obscure usages that may lead to less meaningful sentences.

To improve the perturbation quality, we also allow for a translation-based paraphrasing technique similar to that proposed by Mallinson et al. (2017). Here, we translate the original English sentence into several other languages and then pivot back to English. Provided the translation produces a good result (in our case, we use the Google Cloud

Translation Platform), we can obtain paraphrases of the original sentence (see Figure 2, where the drop-down menu shows some of the perturbed sentences).

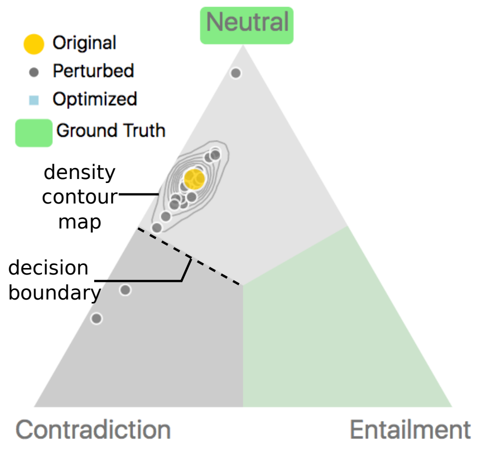


Figure 3: Summary of the prediction results of the perturbed input for the natural language inference model. The prediction is encoded as a point in the barycentric coordinate system of the triangle, in which each vertex corresponds to one prediction label.

3.2 Prediction Visualization

Efficient visual encoding for prediction is crucial for communicating the model behavior, and to fully support the input perturbation feature, the visual encoding should also allow multiple predictions to be shown in the same visualization.

For the natural language inference task, as illustrated in Figure 3, the predicted probabilities are encoded via a triangular barycentric coordinate system, where the vertices represent the three possible predictions (namely, *entailment*, *contradiction*, and *neutral*). The prediction for the original unperturbed input is illustrated by a larger yellow circle, whereas the prediction of perturbed inputs is represented by smaller gray circles.

A density contour of the prediction is computed to emphasize the highly cluttered areas and detect outliers.

3.3 Attention Visualization

As illustrated in Figures 4(a)(b), the most widely adopted visual encodings for attention are bipartite graphs (a) and heatmaps (b). In the graph attention view (Figure 4(a)), the edge thickness corresponds to the attention value of the word pairs. The graph view is suitable for highlighting the most dominant alignments. However, the edges may become cluttered if multiple attention values are high. The matrix attention view (Figure 4(b)) resolves these issues, despite being more verbose and less efficient in highlighting the most dominant alignments. We also enable the linkage between highlighted actions in both views (see Figure 4(a)(b), where one alignment relationship is highlighted).

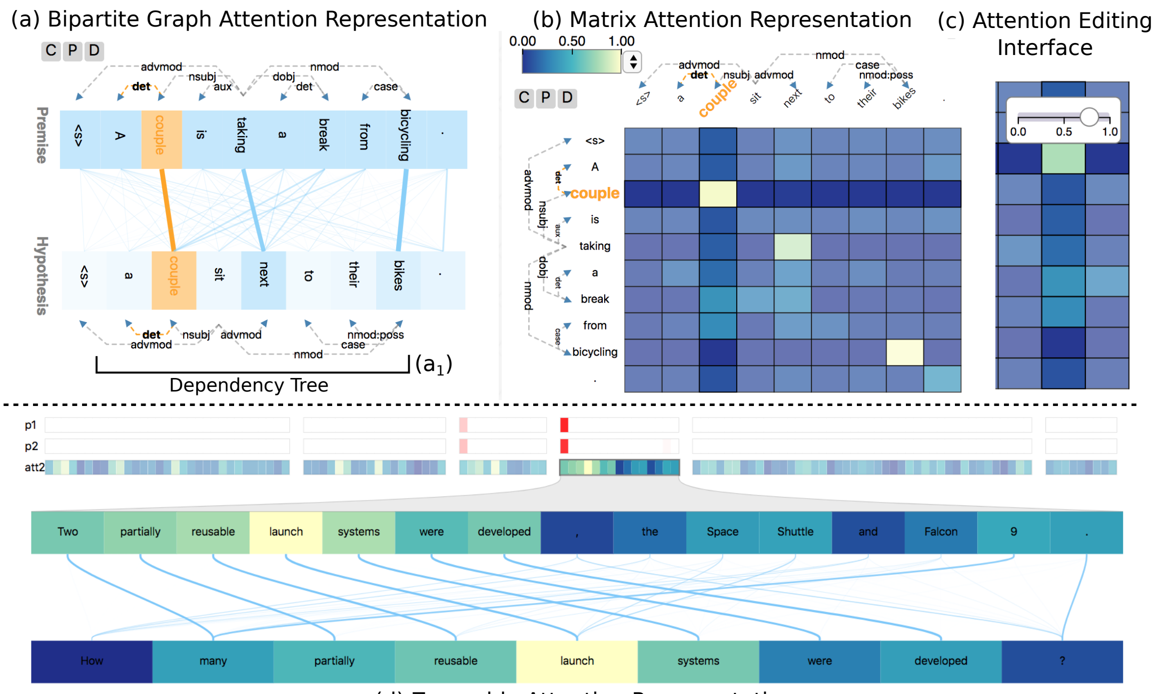
We augment these standard visual encodings with grammar structure to address the challenge of long sentences. The text can be dynamically simplified based on the dependency tree (see Figure 4(a1)). We also show how the dependency information can potentially improve the prediction result in Section 4.1. To facilitate the perturbation of attention (see Figure 1), as illustrated in Figure 4(c), we implement an interface for directly manipulating the attention value.

However, when the text sequence becomes significantly longer, i.e., a full paragraph in the machine comprehension task, even the simplified sentence cannot be meaningfully represented in the graph or matrix visual encoding. To address this visualization challenge, as illustrated in Figure 4(d), we introduce a hierarchical representation. Here, a pure graphical encoding (the color bars marked by *att2*) is used to capture the aggregated attention information for the whole paragraph. The user can focus on localized attention information by selecting a pixel bar that represents a single sentence (the colored blocks in the bar correspond to individual works). We also link this attention representation with the matrix form, such that whenever a sentence is selected the local attention is shown in the matrix view (see Figure 6(a)).

3.4 Implementation

The initial setup cost and unnecessary learning curve are often the barriers to broad adaptation of a tool. Therefore, instead of designing a visualization system as a monolithic standalone application, we implement the proposed system as a Python library with modularity and ease of use in mind. The different pieces of the visualization (i.e., matrix-based attention encoding) can be accessed individually or combined with other components to fit one's workflow via a simple API. More importantly, the library-based design allows easy integration with the existing model implemented in Python. The code for creating an interactive exploration environment for a machine comprehension model is illustrated below.

```
from visPackage import MCModule
```


(a) Bipartite Graph Attention Representation
(b) Matrix Attention Representation
(c) Attention Editing Interface
Premise
Hypothesis
Dependency Tree
(a1)
(d) Zoomable Attention Representation

(d) Zoomable Attention Representation

Figure 4: Attention visualization. A bipartite graph encoding is adopted in the graph attention view (a), in which the edge thickness corresponds to the attention value. The same attention values can also be directly visualized in the matrix form (b). The user can edit the attention values via the pop-up interface illustrated in (c). We overlay the dependency tree (a_1) grammar structure to highlight important words and allow simplification of complex sentences (shown in the video). For highly asymmetric attention, we utilize a zoomable hierarchical visual representation (d). The user can focus on the individual sentence by selecting the summary visualization.

```
from bidaf_src import bidafModelInterface
from NLPutility import translationPerturbation

#initialize machine comprehension model
model = bidafModelInterface(
    wordDict="data/bidaf/squad.word.dict",
    wordVec="data/bidaf/glove.hdf5",
    model="data/bidaf/bidaf.ema")
gen = translationPerturbation()
#visualization components
visLayout = {
  "column":[{"row": ["Paragraph",
                     "AttentionSubMatrix"]},
            {"row": ["AttentionAsymmetric"]}]
  }
#setup interface
modelVis = MCModule(visLayout)
modelVis.setPredictionHook(model.predict)
modelVis.setAttentionHook(model.attention)
modelVis.setSentenceHook(gen.perturbSentence)
#open browser for the web-based visualization
modelVis.show()
```

Listing 1: Code for setting up the visualization system shown in Figure 6(a).

4 Applications

We demonstrate the proposed visualization system on the decomposable attention network (Parikh et al., 2016a) for the NLI task and the BIDAF model (Seo et al., 2017) for the MC task.

4.1 Natural Language Inference

The NLI task predicts the entailment relationship between a premise sentence (P) and a hypothesis sentence (H). The attention matrix captures the alignment of words between these two sentences. Here we give an example of how a wrong prediction can be corrected by editing attention values.

As illustrated in Figure 5(a), the input sentence pair (P:“A woman in a green jacket is drinking tea.” H:“A woman is drinking green tea.”) is predicted to be *entailment*, which is incorrect. By examining the attention, we can see the word *green* in “*green* jacket” is aligned to the *green* in “*green* tea.” However, these two *greens* modify different nouns, which potentially leads to the wrong prediction. The grammatical structure is visually shown in the form of the dependency tree. However, the model does not have access to the syntactic information and mistakenly assumes the two *greens* modify the same thing. To correct the mistake, we can edit the attention value and remove the align between these two “greens” (see (c)(b)). As expected, the prediction label is corrected (*neutral*).

4.2 Machine Comprehension

In the machine comprehension task, the goal is to select a span of text as the answer to a question sentence. The attention information is encoded

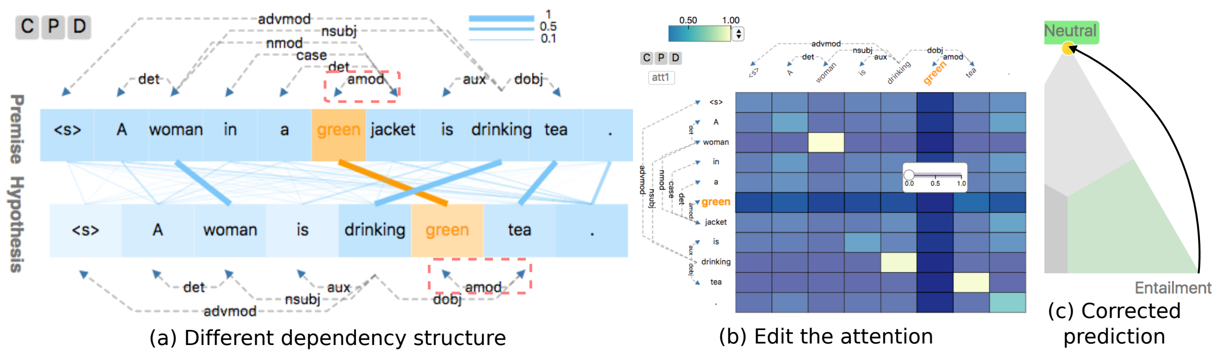
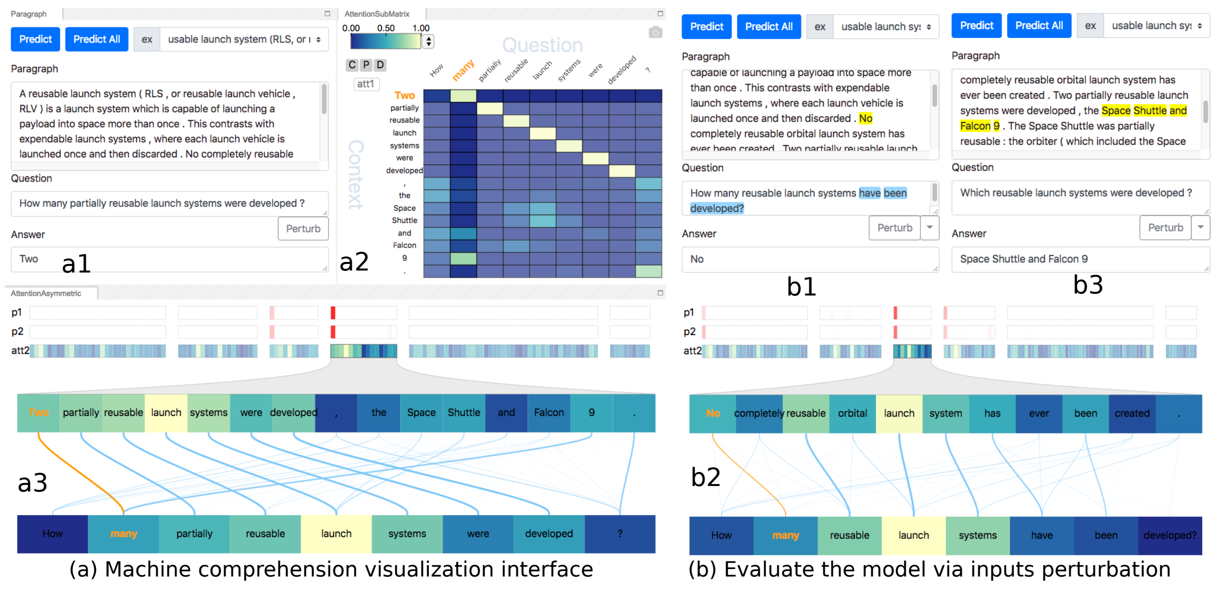


Figure 5: An illustration of the attention editing process. The dependency structure is shown in (a), where the two "greens" decorate different nouns. By removing the "wrong" alignment in (b), the original prediction *entailment* is corrected to *neutral* in (c).


Predict
Predict All
ex
usable launch system (RLS, or r
Perturb
a1
a2
a3
b1
b2
b3
AttentionSubMatrix
AttentionAsymmetric
Question
Context
att1
p1
p2
att2
(a) Machine comprehension visualization interface
(b) Evaluate the model via inputs perturbation

Paragraph
A reusable launch system (RLS , or reusable launch vehicle , RLV) is a launch system which is capable of launching a payload into space more than once . This contrasts with expendable launch systems , where each launch vehicle is launched once and then discarded . No completely reusable

Question
How many partially reusable launch systems were developed ?

Answer
Two

Paragraph
capable of launching a payload into space more than once . This contrasts with expendable launch systems , where each launch vehicle is launched once and then discarded . No completely reusable orbital launch system has ever been created . Two partially reusable launch

Question
How many reusable launch systems have been developed?

Answer
No

Paragraph
completely reusable orbital launch system has ever been created . Two partially reusable launch systems were developed , the Space Shuttle and Falcon 9 . The Space Shuttle was partially reusable : the orbiter (which included the Space

Question
Which reusable launch systems were developed ?

Answer
Space Shuttle and Falcon 9

Figure 6: In the machine comprehension visualization interface (a), the $p1$, $p2$ colored bar (in $a3$) illustrates the predicted start and end index of the answer in the context (the deeper the red, the higher the probability). The most likely answer is shown in ($a1$). The global attention and local attention are visualized by ($a2$, $a3$). We can evaluate the robustness of the prediction by perturbing the question sentence ($b1$, $b3$). As illustrated in ($b1$, $b2$), by removing the word "partial", the model still finds the correct answer (albeit different, as the sentence perturbation changes the exact meaning of the question).

as a bidirectional alignment (i.e., from context to question and vice versa). Here, we refer the context to question attention as *att1* and the question to context attention as *att2* (which is a vector instead of a matrix according to Seo et al. (2017)). In this demo, we apply a min max normalization for *att2* after the softmax layer to better distinguish different attention values.

As illustrated in Figure 6, we represent *att2* as colored bars with a *yellow-green-blue* colormap. Each rectangular bar corresponds to one sentence. The user can focus on individual sentences by clicking on the rectangular bar (see Figure 6). The $p1$, $p2$ colored bars (*white-red* colormap) illustrate the predicted probabilities of the start and the end index for the answer (the deeper the higher). In Figure 6(a3), the sentence containing the answer exhibits good alignment with the question (e.g., "Two" with "many"). Interestingly, the number "9" (in "Falcon 9") is also aligned with "many", which may lead to problems.

The user can explore the robustness of the model by examining how the prediction varies when the question is perturbed. As illustrated in ($b1$, and $b2$), the perturbation removes the word "partial" in the original sentence, which leads the model to produce a different yet correct answer ("No"). Referring to ($a1$), we can see the word "No" exhibits the second highest probability for the original question. The user can also manually edit the text. As shown in ($b3$), the model still produces the correct answer when changing the question from "how many" to "which".

5 Discussion

In this work, we introduce a visualization library for creating customized environments that allows the user to interrogate the relationships among different parts of the model pipeline via interactive queries. We demonstrate the usability and flexibility of the tool by configuring the visual components to investigate models for different NLP tasks (e.g., NLI, MC). We also conducted a small-scale user evaluation, in which five Ph.D.-level students with an NLP background spent 30 minutes with the tool and then provided feedback. Most suggested the environment allowed them to refine queries iteratively and identify potential issues with the model, but some also mentioned the tool might not provide enough guidance for users who do not have an in-depth understanding of the model at hand.

Even though we designed the individual components with versatility in mind, due to a large number of variants of attention networks, we found it difficult to ensure compatibility with all the available configurations. In the future, we plan to improve upon the current attention interface, release the library as an open-source package, and expand the visualization components to handle tasks such as neural machine translation and more.

Acknowledgments

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. This work is also supported in part by NSF: CGV: Award: 1314896, NSF: IIP: Award: 1602127, NSF: ACI: award: 1649923, DOE/SciDAC DESC0007446, CCMSC DE-NA0002375, PIPER: ER26142 DE-SC0010498, and NVIDIA Corporation. This material is based upon work supported by the Department of Energy, National Nuclear Security Administration, under Award Number(s) DE-NA0002375.

References

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

Jaesong Lee, Joong-Hwi Shin, and Jun-Seok Kim. 2017. Interactive visualization and manipulation of attention-based neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 121–126.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2015. Visualizing and understanding neural models in nlp. *arXiv preprint arXiv:1506.01066*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 881–893.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016a. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. Association for Computational Linguistics.

Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016b. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

Andreas Rücklé and Iryna Gurevych. 2017. End-to-end non-factoid question answering with an interactive visualization of neural attention weights. *Proceedings of ACL 2017, System Demonstrations*, pages 19–24.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. *ICLR*.

Hendrik Strobelt, Sebastian Gehrmann, Hanspeter Pfister, and Alexander M Rush. 2018. 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.

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 6000–6010.

Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.