Skylar Wurster

CSE 5544

Professor Chen

Jan 31st, 2019

Assignment 2

[DeepCompare: Visual and Interactive Comparison of Deep Learning Model Performance](http://sugeerth.cs.ucdavis.edu/Papers/deepcompare.pdf); Sugeerth Murugesan, Sana Malik, Fan Du, Eunyee koh, Tuan Manh Lai; VDS 2018

DeepCompare aims to improve on traditional methods of comparing deep learning models by aiming to show qualitative characteristics of why a model might outperform another model for some subset of the input domain. Sugeerth Murugestan from UC Davis, Sana Malik, Fan Du, and Eunyee koh from Adobe Research, and Tuan Manh Lai from Purdue University offer a visual analytic approach called DeepCompare to assess these tradeoffs.

The team began their work with the intention of illuminating to humans how very complex models are currently working in comparison to each other. They note that the advent of deep learning and the usefulness of extremely large models, it has become significantly more difficult to interpret what they are doing, and how they can be improved. Other methods exist to dissect a single model, but none to compare multiple. This is especially useful for comparing networks with different architectures altogether.

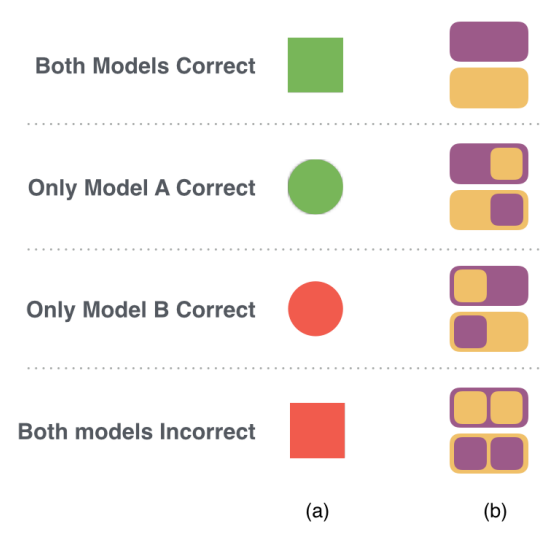
It is worth noting that these researchers interview five machine learning practitioners to find out what kind of tool would be useful for them. Since I am not a machine learning practitioner myself, I will not make any suggestions for improvement upon the three tasks and sex challenges the writers design from their interview with the researchers.



This visualization is comparing two models, a CNN and an LSTM, both trained on the same data and with accuracies within 0.2%. Panel (a) is the Neuron Weights Detail panel, which shows the weights of one layer of the model as a heatmap, where green is higher activation, and red is lower activation. We can see from this image that the CNN is set to layer 5 and the LSTM is set to layer 3. If a single node is selected from panel (a), panel (b) populates with data about that specific neuron. The data here shows the neuron activation distribution through all the times the trained model was run. Panel (c), named the Test Result Detail panel, lists each of the inputs in order of which the selected neuron fired the most. In this panel we also see a comparison of the two models for that input. Under “result”, we see a background color (yellow or purple), and up to two squares on either the left or ride side of the background. The background color indicates the ground truth, the left square indicates the CNN’s result, and the right square indicates the LSTM’s result. These inputs can also be filtered to show items that passed or failed in certain circumstances. Lastly, panel (d) labeled Test Result Summary shows an overview for both models. It breaks the input dataset into negative (yellow) and positive (purple). From there, area is used to show correctness for both models within the negative and positive sentiments.

Overall, we see good use of Schneiderman’s “Overview first, zoom and filter, then details on demand”. Panel (d) offers a high-level overview of the accuracy of each model on the test set used. Color is used to distinguish between the negative and positive ground truths, accompanied by a legend above. The panel uses area to depict a proportion, which isn’t a good choice. We see that it is difficult to see whether the CNN correct in the negative sentiment is the smaller or larger than “both incorrect” in the positive sentiment side. A bar chart, which utilizes length, would better convey this piece of quantitative data. Two bar charts could be made, one with purple bars for the positive sentiments, and one with yellow bars for the negative sentiments.

Moving down and filtering, we come to panel (c). Since the actual input sentence fragments are more understandable than individual neurons, filtering by the fragments might be the next step a researcher would want to look at. The visualization gives us filter options to find instances of one model being correct and the other one being incorrect, vice versa, or where they’re both incorrect, or both correct. This coloring system went through iterations, and the researchers decided to scrap their previous glyph system which was misleading due to adding shape when unnecessary. Below shows the old glyph system on the left, and the new coloring on the right.



When clicking on a sentence fragment in (c), the neurons in panel (a) align from most activated to least activated for that given input. This intractability allows for the users to find trends in the data that is difficult for an algorithm to point out.

Following that, panel (a) works in much the same way as panel (c). Clicking on a neuron rearranges panel (c) to show which inputs cause that neuron to activate the most, giving a great deal of information to the user. For instance, one node might respond heavily to Spanish fragments. This is exactly “details on demand”.

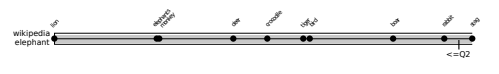
Panel (a) is the most cluttered with numbers that often don’t have labels. Initially, it must be explicitly told to the user or inferred that the squares represent neurons. Further, the positioning of these squares is confusing. The squares are top left aligned, so smaller squares appear in the top left of the character box. This causes awkward spacing between columns that might be interpreted as something significant from a user. If these squares were center aligned, this issue would disappear. We also see a lack of a scale for size of each square. A color gradient exists for activation power, but no legend exists to describe what the size means. Additionally, since these depict quantitative data, it is curious that they have chosen to use both color and area, which are the two worst attributes for quantitative data, and not length or angle/slope. We also have no visual indication as to what the two rows are corresponding to in this image.

Overall, the conclusion of the paper shows useful results from the machine learning practitioners interviewed for this software. Improvements can be made specifically to panel (a) and (d). Panel (a) takes up most of the room but doesn’t do much with the space. Also, the order of the panels isn’t confusing, but doesn’t lend itself to the overview, filter, details process. Colors are used intelligently, and the overall aesthetic is nice.

[Interactive Analysis of Word Vector Embeddings](https://graphics.cs.wisc.edu/Papers/2018/HG18/embeddings_preprint.pdf); F. Heimerl and M. Gleicher; (EuroVis) 2018

F Heimerl and M Gleicher create a visualization tool for word embeddings to give practitioners the ability to investigate neighborhoods and reconstructed co-occurrences and aligning word vectors based on concept axes. These goals were based on the literature in the domain. After creating their visualization, domain users gave feedback for its usefulness.

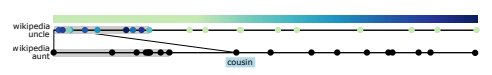
One current method for this area of visualization is the EnsembleMatrix approach, which allows users to combine multiple single models to improve quality. This has been extended upon to allow users to interact with and modify one model over many iterations. Another method to visualize this embedding is through dimensionality reduction, such as t-SNE. Other works mainly visualize the training process, but not the finished embedding.



Above is the visualization shown for local neighborhoods – specifically the 10 nearest neighbors for the word “elephant” on Wikipedia. The goals for this visualization are that users should be able to view the closest neighbors of a word’s vector and generalize the distances to others. The cosine distance is used as this is the most common choice in the literature. This leads to misleading information, as the only distance that is correct is the distance to the word in question. Relative distances between any other pair of words on the plot are incorrect. This can make two words look more similar than they are in the vector space.

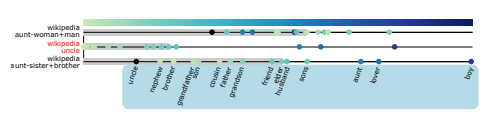
The authors make an interesting decision to avoid the 2D vector embedding through methods such as PCA, MDS, and t-SNE. They are cautious about the noise from the actual methods of dimensionality reduction and decide to stick with a 1D plot. I agree with this decision, as it removes another dimension of information that viewers must process.

There is also interaction for this tool. Hovering over the gray padding behind the axis will show a tool tip with the quartile of neighborhood sizes in the embedding it falls into. This interaction technique allows specialists to gather information about how meaningful the relationship in a neighborhood is, since the above is quartile based so something with a large distance can still appear very close to the word in question.

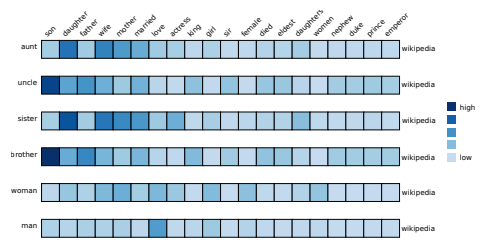


A similar method is used when extending the visualization to allow the comparison of multiple neighborhoods. The shaded gray highlights the nearest n neighbors for each word, where the user choose their n. A color ramp is used here to show the position of each word on the axis following the current word. This helps depict relative distances. Users may click on any of the dots to reveal the word as well as a link to the word on the other axis, if present. Above, cousin is selected, and we can see it is closer to “uncle” than “aunt” in this embedding.

I believe the color ramp to be somewhat confusing here. Since quantifiable data is used, it might be better to use something more representative than color. The authors point this out but defend by saying that the changes in relative position may be easily discerned as breaks in the color gradient. Additionally, the interactivity in this example seems as though it may be clunky; if two words are overlapping each other, it may be impossible to select one of the words with the cursor. Another design decision is to hide the words by default due to too much clutter on the screen with them. If this is the case, I believe another visualization might work better.

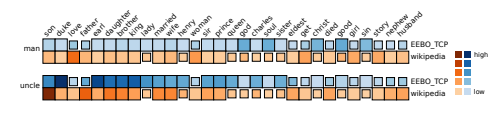


Another visualization compares multiple local neighborhoods. All the same visualization techniques are used as before, except now the user is hovering over an axis, so all words are displayed.

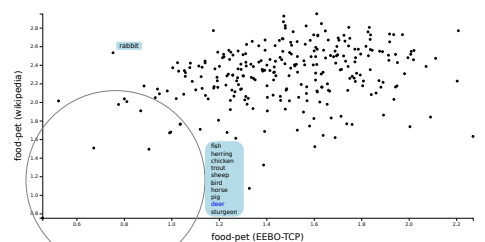


This next visualization displays the co-occurrences of words using a gradient heatmap. The authors note that other designs involving bar charts were tested, but that this solution allows for easier comparisons and pattern recognition to human eyes. A tree-based design was also tested but was not useful for the authors scenarios.

For the above design, users must choose the words they with to investigate. Then, the rows and columns are determined, and visualization created. This is great for an overview but doesn’t allow and further analysis.



The same tool may be used to analyze multiple embeddings on different datasets. The intensity of the hue chosen is the co-occurrence of those two words. Since different datasets might not have any co-occurrence of words, it is denoted by making the box smaller. For instance, in the Wikipedia dataset, there is not co-occurrence of “man” with “good”, though there is a somewhat strong co-occurrence in the EEBO\_TCP dataset.



Another tool for comparing embeddings is by selecting two words and analyzing local neighborhoods of each within each embedding, plotted in 2D with likeness to each embedding on each axis. If the data has a linear trend, then the embeddings are more similar, which allows for a nice overview. Interaction can be done by creating a circle at a given point and seeing all words within that area. This allows for details on demand.

The visualization techniques proposed in this paper do a very good job a comparing multiple embeddings, but not as well for single embeddings. If there were more connectivity between each visualization, then this would be a very smooth and helpful tool but seems to feel clunky and disjoint.