Visualizing Effects of Hyperparameters in NLP Neural Models

Robin Lashof-Regas
Robin.Lashof@Berkeley.edu

Abstract

Ever-more complex neural network architectures have established themselves as clear choices for common NLP tasks such as sentiment analysis, machine translation, and more. However, a large challenge that these models pose is their explainability. In particular, it is usually unclear why specific choices in model hyperparameters perform better than others. In this work, we focus on dimensionality of layers in a bidirectional LSTM model for sentiment classification. We use state of the art techniques in model visualization to understand how changes to the layer dimensions affect the accuracy of the model, with a specific focus on the the ability to learn compositionality. Through this analysis we show the differences in characteristics between embedding dimensions and dimensions of LSTM layers.

1 Introduction

Complex neural architectures such as LSTMs (Hochreiter and Schmidhuber, 1997) have become the norm for machine learning tasks in NLP and beyond. A major limitation of these models, however, is their explainability. While significant work has been done to implement and analyze various techniques for explaining model predictions in NLP (Li et al., 2015; Arras et al., 2017a), these works do not investigate the effect of model hyperparameters on those explanations. Other work focuses on hyperparameter optimization, mostly using techniques like random search, i.e. smart guess and check, which don't provide motivation for those hyperparameter choices (Bergstra et al., 2011).

In this work we investigate the effect of a wide array of hyperparameter choices for the bidirectional LSTM from Li et al. (2015), focusing on the dimensionality of the embedding and hidden layers. First, training the LSTM on a span of choices for layer dimensions, we evaluate the accuracy of the resulting model as a function of each hyperparameter. Next, applying the layer-wise relevance propagation technique described in Arras et al. (2017b) we focus on a hand-selected set of test sentences that exemplify compositionality, and compare the accuracy of each model on this set. Finally, we evaluate each model against a set of short test phrases and contrast the predictions of each model, focusing on the prediction score for the true class.

2 Background

While there has been work to understand how to best choose number of nodes and layers in neural networks (Stathakis, 2009; Bergstra et al., 2011), the methods used are primarily search-based and do not provide intuitive explanations for the choices they make. In order to provide these intuitive explanations, we need to understand both why and how neural networks make the predictions they do.

In pursuit of the ”why”, there have been significant advances in the area of neural network explanation techniques. Simonyan et al. (2013), for example, proposed the method of using saliency maps in image classification to visualize gradients of output class scores with respect to the model inputs. Li et al. (2015) later adapted these methods for NLP. More recently, a new method known a layer-wise relevance propagation (LRP) was developed to explain the predictions of deep neural networks for image classification (Bach et al., 2015). This method was later adopted to work for NLP tasks and then expanded to work for RNNs (Arras et al., 2017a; Arras et al., 2017b). Additional work has also been done to compare these various techniques for NLP applications which concluded that LRP with the specific rule for multiplicative connections proposed by Arras et al. (2017b) produced the best results (Arras et al., 2019). For this reason, we focus on this ”LRP-all” method for visualizing the neural network predictions in this paper.

To our knowledge there has been little work done to explain the how, i.e. how the structure (layers and nodes) of the network influences it's predictions. This represents the primary topic for this paper.

3 Methods

In this section we discuss the following: (1) the dataset and task used for both training and evaluation, (2) the architecture of the model in use, and (3) the LRP-all visualization technique used to evaluate predictions.

3.1 Dataset and Task

The dataset and task used for training is the benchmark 5-class sentiment classification of movie reviews over the Stanford Sentiment Treebank (Socher et al., 2013). While there are two evaluation metrics on this task: the full 5-class ”fine-grained” (very positive, positive, neutral, negative, very negative) and the partial ”coarse-grained” (positive, negative), training is performed on the fine-grained task.

3.2 Model

The model used in our analysis is the bidirectional LSTM described in Li et al. (2015). The model uses a single hidden layer to learn embeddings followed by a standard bidirectional LSTM architecture. Finally, the last node of both the left and right LSTMs is fed into a linear output layer that then makes predictions using a softmax classifier. Training is done with AdaGrad using mini-batch with dropout set to 0.2. The modification we make to this network is to separate the dimension used in the embedding and the dimension used in the rest of the network; we denote these as the embedding dimension (d_e) and hidden dimension (d_h) respectively. This allows us to vary these two dimensions independently in order to analyze how the model accuracy changes with respect to each.

3.3 Layer-wise Relevance Propagation

In order to produce the visualizations and relevance scores presented in this work we utilize layer-wise relevance propagation with the ”source takes all” rule for multiplicative connections (Arras et al., 2017b). LRP relies on the simple ”conservation of relevance” rule that $\sum R_i = \sum R_j$ where the sums represent the sum over relevances of each node at any two of the layers in the network. Simple weighted connections use a relevance propagation rule of the following form:

$$R_{i \leftarrow j} = \frac{z_i w_{ij} + \frac{\epsilon sign(z_j) + \delta b_j}{N}}{z_j + \epsilon sign(z_j)} R_j \qquad (1)$$

where z_i represents the activated value of the node being updated, and $R_i = \sum_j R_{i \leftarrow j}$ where j represents the index of nodes in the next layer after i.

The intuition for equation (1) is that the portion of the relevance of a node i contributed by a node j of the next layer is proportional to the weight between the two nodes times the ratio of their activated values (with fudge factors to avoid division by zero). For LSTMs with multiplicative connections combining a ”gate” node with a ”source” node it has been shown that assigning $R_{gate} = 0$ and $R_{source} = R_j$ seems to produce the best results (Arras et al., 2019).

By assigning the relevances of the output layer to be a one-hot encoding on the target or predicted class (depending on what is being analyzed) one can apply the rules above recursively to obtain the relevances of each dimension of the input.

4 Results and Discussion

In this section we describe our results. To start, we discuss the effect of varying the embedding and hidden dimensions on the fine and coarse-grained accuracy of the model against the test set. Next, we analyze the models against a set of hand-selected test sentences that display difficult to model attributes of compositionality such as concession and negation. Finally, we use a toy example to show how the different models handle simple negation in an attempt to understand the results from the other, more broad analyses.

4.1 Dimension vs Accuracy

As a baseline, we begin by using the same parameters described in Li et al. (2015), namely $d_h = d_e = 60$. With these parameters we achieved a fine-grained accuracy of 0.473 and coarse-grained accuracy of 0.839 on the test set. With this baseline, we proceed to vary our hidden and embedding dimension hyperparameters in order to model

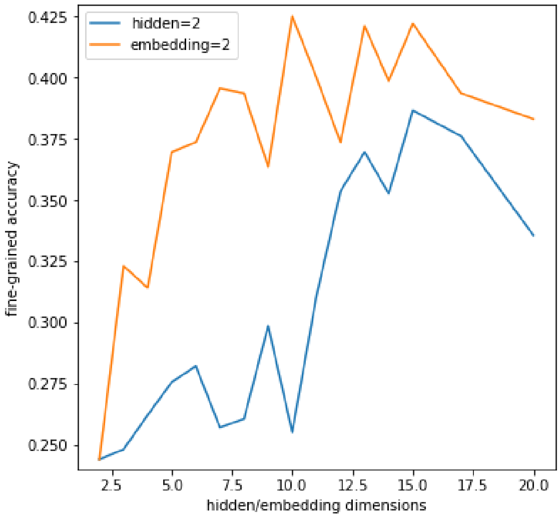


Figure 1: Fine-grained accuracy of model w.r.t. hidden/embedding dimensions while keeping the other dimension constant at 2.

their relationships with the two accuracy measurements.

In figure 1, we show a plot of the fine-grained accuracy of the model with constant hidden/embedding dimension set to 2. There is a clear difference between the resulting model for small hidden vs embedding dimensions. This tells us two things: (1) the embedding vs hidden layers capture significantly different information (if they did not, the lines would look the same) and (2) it is ”easier” for the model to overcome a smaller embedding dimension vs a smaller hidden dimension.

To dive deeper into this let's investigate some examples that one model predicted correctly but the other did not. In the first example in figure 2 we show a sentence that was classified correctly the majority of the time when d_e was held at 2, and was classified incorrectly when d_h was held at 2. In this case, the LRP heatmap gives us a lot of insight into what's going on. In the case where $d_e = 2$ we can see that the word 'but' is given very high relevance in favor of the predicted class, and with $d_h > 2$ the sample is classified correctly. Additionally, when $d_h = 2$, the word 'but' is given less relevance as, the model puts a higher reliance on the learned sentiment of individual words vs the compositionality of the sentence. This is an important result as it indicates that there may be a ”gate” in the number of nodes required by the hidden layers in order to learn certain aspects of compositionality. While this example is very low dimension, we expect this to be true in higher dimensions as well.

In the second example in figure 2 we show the opposite case - the model with a low embedding dimension is not able to predict the correct class. Here we can attribute this to the fact that the model is not able to learn the relative importance of the word 'tedious' compared to the word 'fun' when the embedding dimension is small.

4.2 Compositionality Examples

In order to understand the relationship between hyperparameters and the ability to learn compositionality we introduce a hand-picked set of test sentences that exemplify the complex relationships we are trying to test. In particular we focused on sentences from the test set that contain the words ”but”, ”though”, or ”n't”, and then manually selected 212 sentences from this set where the correct class was highly dependent on the value added by these words.

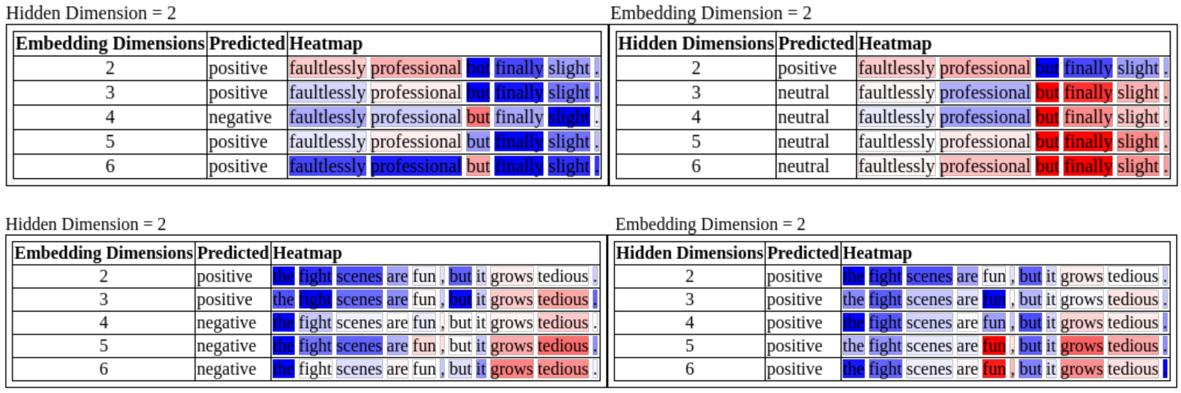
The first thing we notice is that the coarse accuracy of the base model on this test set is much lower, while the fine accuracy is similar. In particular, with $d_h = d_e = 60$ the accuracies were 0.472 (fine) and 0.608 (coarse) as opposed to 0.473 and 0.839 on the full test set. This gives evidence that we chose a good set of test sentences as the smaller gap between fine and coarse grained accuracy is an indication that when the model correctly interprets the compositional words, obtaining the correct coarse-grained sentiment, it is able to determine the true class more often.

Another interesting observation is that for this test set we do not see the same stark contrast between the fine-grained accuracy of small hidden dimensions vs small embedding dimensions, as seen in figure 3. This result indicates that the hidden and embedding dimensions play relatively equal roles in cases of complex compositionality.

4.3 Toy Example

In this section we evaluate a toy example and show an interesting result with respect to the relationship between the model dimensions and the true class score.

The example we cover is the simple negation ”not good”. The true class for this example is negative. The interesting result that we find is that as the hidden dimension of the model increases, the relationship between the embedding dimensions and the softmax score for the true class with constant hidden dimension approaches a linear fit as



Hidden Dimension = 2

Embedding Dimensions	Predicted	Heatmap
2	positive	faultlessly professional but finally slight .
3	positive	faultlessly professional but finally slight .
4	negative	faultlessly professional but finally slight .
5	positive	faultlessly professional but finally slight .
6	positive	faultlessly professional but finally slight .

Embedding Dimension = 2

Hidden Dimensions	Predicted	Heatmap
2	positive	faultlessly professional but finally slight .
3	neutral	faultlessly professional but finally slight .
4	neutral	faultlessly professional but finally slight .
5	neutral	faultlessly professional but finally slight .
6	neutral	faultlessly professional but finally slight .

Hidden Dimension = 2

Embedding Dimensions	Predicted	Heatmap
2	positive	the fight scenes are fun , but it grows tedious .
3	positive	the fight scenes are fun , but it grows tedious .
4	negative	the fight scenes are fun , but it grows tedious .
5	negative	the fight scenes are fun , but it grows tedious .
6	negative	the fight scenes are fun , but it grows tedious .

Embedding Dimension = 2

Hidden Dimensions	Predicted	Heatmap
2	positive	the fight scenes are fun , but it grows tedious .
3	positive	the fight scenes are fun , but it grows tedious .
4	positive	the fight scenes are fun , but it grows tedious .
5	positive	the fight scenes are fun , but it grows tedious .
6	positive	the fight scenes are fun , but it grows tedious .

Figure 2: LRP heatmaps for example sentences that were predicted correctly the majority of the time with one dimension held constant but incorrectly with the other. Red indicates a positive contribution to the predicted class whereas blue represents a negative contribution. The true class is neutral for the first example and negative for the second.

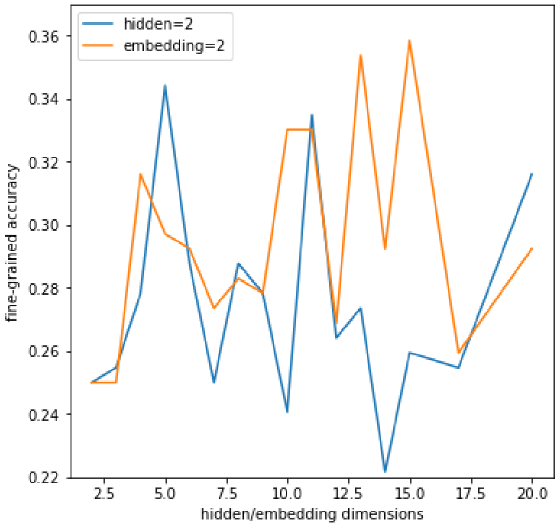


Figure 3: Fine-grained accuracy of model on hand-picked sentences displaying complex compositionality w.r.t. hidden/embedding dimensions while keeping the other dimension constant at 2.

seen in Figure 4.

With $d_h = 20$ and d_e ranging from 2 to 20, the linear regression line shown in Figure 4 has an R^2 of 0.944. This is very interesting as it exemplifies an interdependency between the hidden dimensions and the embedding dimensions, i.e. as the hidden dimension approaches a certain threshold, each increase in the embedding dimension has a greater effect in influencing the model's chance in predicting the true class.

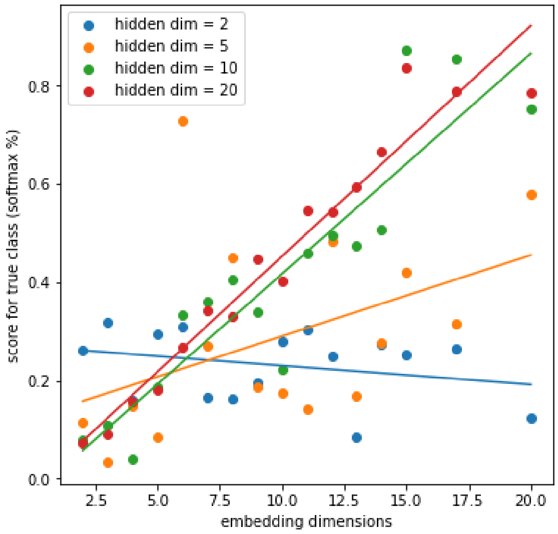


Figure 4: Relationship of embedding dimension to softmax score for the true class at constant hidden dimension for the toy phrase of ”not good”.

5 Conclusion

In this paper, we have shown significant progress in understanding the effects on model predictive power of varying the number of nodes in both embedding and hidden dimensions. While much of our analysis is in very low dimension, the concepts described here illustrate relationships that may hold true in higher dimensions and more complex models (albeit to a lesser extent and in a more intricate way).

We have demonstrated in multiple scenarios that the information captured by the model hidden and embedding dimensions is fundamentally different. We have also shown that understanding complex compositionality relies significantly on the values of both of these parameters whereas good classification on the larger test set can usually be achieved with one or the other (for example, with $d_h = 3$ and $d_e = 13$ we achieved a coarse-grained accuracy of 0.821).

Finally, we showed that in a toy example the classification probability for the true class becomes linear in the dimension of the embedding (for small d_e at least) as the hidden dimension increases. The importance of this result is not the linear nature, rather the existence of any relationship that can be modeled, suggesting that the same may be possible in more complex and higher dimensional scenarios. Further research is needed to evaluate this possibility.

References

Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2017a. "what is relevant in a text document?": An interpretable machine learning approach. *PLOS ONE*, 12(8):e0181142, Aug.

Leila Arras, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2017b. 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, Copenhagen, Denmark, September. Association for Computational Linguistics.

Leila Arras, Ahmed Osman, Klaus-Robert Müller, and Wojciech Samek. 2019. Evaluating recurrent neural network explanations.

S Bach, A Binder, G Montavon, F Klauschen, K-R Müller, and W Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):e0130140.

James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2546–2554. Curran Associates, Inc.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2015. Visualizing and understanding neural models in nlp.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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, Seattle, Washington, USA, October. Association for Computational Linguistics.

D. Stathakis. 2009. How many hidden layers and nodes? *International Journal of Remote Sensing*, 30(8):2133–2147.