Understanding Deep Learning Performance through an Examination of Test Set Difficulty: A Psychometric Case Study Supplemental Material

A DNN Model Architecture

Here we provide a brief overview of the model architectures for the deep neural network (DNN) models used in our experiments. For additional details we refer the reader to the original papers.

A.1 Long Short Term Memory

The Long Short Term Memory (LSTM) model used here was provided by (Bowman et al., 2015) with the release of the SNLI corpus. The model consists of two LSTM sequence-embedding models (Hochreiter and Schmidhuber, 1997), one to encode the premise and another to encode the hypothesis. The two sentence encodings are then concatenated and passed through three tanh layers. Finally, the output is passed to a softmax classifier layer to output probabilities over the task classes. For SA, we kept the same architecture but used a single LSTM layer to encode the input text.

A.2 Convolutional Neural Network

We used the convolutional neural network (CNN) model of (Kim, 2014) in our experiments. For each input, the word vector representation of the input tokens were concatenated together to form a matrix. A series of convolutional operations were applied, followed by a max-pooling operation and a fully connected softmax classifier layer. More concretely, for an input sentence $\mathbf{x}$, let $\mathbf{x}_i$ be the word vector representation of the i-th word in $\mathbf{x}$. The convolution operation of filter $\mathbf{w}$ over a window of length h starting with word $\mathbf{x}_i$ results in a context vector c_i:

$$c_i = f(\mathbf{w} \cdot \mathbf{x}_{i:i+h-1} + b) \tag{1}$$

where b is a bias term (Kim, 2014). The filter is applied over all windows in the sentence to generate a feature-map, and max-pooling is used to identify the feature for this particular filter. The process is repeated with multiple filters, and the output features are then passed to a softmax classification layer to output probabilities over the class labels (Kim, 2014). For NLI, the premise and hypothesis sentences were concatenated before encoding.

A.3 Neural Semantic Encoder

Neural Semantic Encoder (NSE) is a memory-augmented neural network that uses *read*, *compose*, and *write* operations to evolve and maintain an external memory (Munkhdalai and Yu, 2017):

$$o_t = f_r^{LSTM}(x_t) \tag{2}$$

$$z_t = softmax(o_t^\top M_{t-1}) \tag{3}$$

$$m_{r,t} = z_t^\top M_{t-1} \tag{4}$$

$$c_t = f_c^{MLP}(o_t, m_{r,t}) \tag{5}$$

$$h_t = f_w^{LSTM}(c_t) \tag{6}$$

$$M_t = M_{t-1}(\mathbf{1} - (z_t \otimes e_k)^\top) + (h_t \otimes e_l)(z_t \otimes e_k)^\top \tag{7}$$

where f_r^{LSTM} is the read function, f_c^{MLP} is the composition function, f_w^{LSTM} is the write function, M_t is the external memory at time t, and $e_l \in R^l$ and $e_k \in R^k$ are vectors of ones (Munkhdalai and Yu, 2017).

For NLI, the premise and hypothesis sentences were each encoded with an NSE module. The outputs were combined and passed through a softmax classifier layer to output probabilities. For SA, we kept the same architecture but used a single NSE layer to encode the input text.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and D. Christopher Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642. Association for Computational Linguistics.

S Hochreiter and J Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Association for Computational Linguistics.

Tsendsuren Munkhdalai and Hong Yu. 2017. Neural semantic encoders. *EACL 2017*.