Learning sentence representations from natural language inference data

Course Advanced Topics in Computational Semantics
Students Lisa Saleh (12613940)
Deadline April 22 (2025), 23:59

1 Introduction

This practical focuses on learning sentence representations from Natural Language Inference (NLI) data. The goal is to implement and evaluate several sentence encoder models trained on the Stanford Natural Language Inference (SNLI) dataset. This practical partially replicates the work of Conneau et al. [3]. Sentence encoders map variable-length sentences into fixed-size vector representations, which can be used for classification as well as transfer to downstream tasks.

To evaluate the quality of the learned representations, I use two settings:

- **In-domain evaluation** on the SNLI dataset [1], using a classifier to predict the entailment relation (entailment, neutral, contradiction) between a premise and a hypothesis.
- **Transfer evaluation** using the SentEval benchmark [2], which tests whether the learned sentence vectors generalize to other tasks such as sentiment analysis, paraphrase detection, and question classification.

I implemented four models, following the architecture described by Conneau et al. [3]. Each of these offers different trade-offs in terms of complexity, representation quality, and ability to generalize:

(a) **Baseline:** Averages pretrained GloVe word embeddings across the sentence. This model is extremely fast and simple, but discards word order and syntactic structure. It often performs better on classification tasks where lexical content is sufficient, but lacks the capacity to capture compositional meaning or sentence structure.

(b) **LSTM:** A unidirectional LSTM that returns the final hidden state as the sentence representation. This model captures word order and can model sentence-level dependencies. However, it tends to prioritize information at the end of the sequence and may struggle with long-range dependencies in earlier parts of the sentence.

(c) **BiLSTM:** Combines the final hidden states of both a forward and backward LSTM. This setup allows the model to incorporate contextual information from both directions of the sentence, improving representation quality over the unidirectional variant. Still, using only the final states may result in loss of information from the middle of the sentence.

(d) **BiLSTM-Max:** Computes hidden states from a bidirectional LSTM and applies max pooling over the entire sequence to form the sentence vector. This architecture is often the strongest among the four, as it extracts the most informative features from the entire sentence, regardless of position. This makes the model more robust to changes in word order and especially good at capturing strong lexical cues. However, it may discard subtle interactions between words that do not result in high activations for individual units.

Each sentence encoder produces a vector representation for both the premise and the hypothesis. These two vectors are then combined using three operations: concatenation, element-wise difference, and element-wise product. The resulting combined vector is passed through a multi-layer perceptron (MLP), which outputs a prediction for the relationship between the two sentences (entailment, contradiction, or neutral).

2 Methods

2.1 Training Setup

All models were implemented in PyTorch and trained on the SNLI dataset. I lowercased all input and used NLTK's TreebankWordTokenizer for tokenization. For word embeddings, I used the 300-dimensional GloVe vectors trained on

Common Crawl (840B tokens). These embeddings were fixed during training and aligned to a vocabulary built from the SNLI training split.

I trained each model for 20 epochs using stochastic gradient descent (SGD) with a learning rate decay of 0.99 applied after each epoch. Early stopping was triggered when the learning rate dropped below 10^{-5}. Training and evaluation were performed on Snellius, using the full training data. A smaller debug subset was used during development to verify the implementation.

2.2 Design Choices

A key implementation detail was aligning the vocabulary used in the dataset with the GloVe embeddings. I chose to build the vocabulary solely from the SNLI training set. This simplified the setup and ensured that the embedding matrix only contained tokens present in the training data. After alignment, the embedding matrix was initialized and kept fixed throughout training.

The LSTM-based model includes a dropout layer with a dropout rate of 0.2. I added this after observing that the model struggled to converge during early training runs. Dropout helped regularize the network and improved training stability. The BiLSTM and BiLSTM-Max models did not include dropout, as they performed well without it in preliminary experiments.

3 Results

Table 1 summarizes the performance of all four models on the SNLI dataset and SentEval transfer tasks. Both SNLI development are reported and test accuracy, along with micro and macro averaged accuracy across seven SentEval tasks. BiLSTM-Max achieves the best results across all metrics.

Model	SNLI Dev (%)	SNLI Test (%)	SentEval Micro (%)	SentEval Macro (%)
Baseline	74.80	74.79	80.72	79.84
LSTM	81.38	81.18	77.19	76.28
BiLSTM	82.48	82.32	80.33	79.29
BiLSTM-Max	85.51	84.44	83.21	82.11

Table 1: SNLI classification accuracy and SentEval transfer accuracy for each model.

4 Analysis and Discussion

4.1 Analysis of Results

The BiLSTM-Max model outperforms all others on both SNLI and SentEval, achieving 85.51% on SNLI dev and 83.21% SentEval micro accuracy. Performance rankings are consistent across both evaluation settings: as model complexity increases, so does performance.

These trends align well with the findings of Conneau et al. [3]: max pooling significantly boosts transfer performance, and bidirectional encoders provide gains over unidirectional ones. My absolute scores are slightly lower than those reported by Conneau et al. [3], likely due to limited training time and a lack of hyperparameter tuning.

Interestingly, the baseline outperforms the LSTM on both SNLI and SentEval, which is unexpected. I suspect this may be due to overfitting in the LSTM, especially given the absence of regularization and tuning.

Overall, I find that the simpler baseline and LSTM models underperform but train quickly. BiLSTM and BiLSTM-Max benefit from richer sentence encoding via bidirectionality and pooling. The performance gap between models is larger on SNLI than on SentEval, suggesting that SNLI is more sensitive to encoder architecture than general transfer tasks.

4.2 Error Analysis

In this section, I analyze a few model predictions to better understand common successes and failure modes. Overall, the models perform reliably when semantic relationships are clear and straightforward. However, they struggle with negation, logically unrelated statements, and require more world knowledge than static embeddings can provide. These challenges are typical for models without attention or deeper semantic modeling, and highlight the limitations of purely surface-level lexical reasoning.

4.2.1 Correct Predictions

Premise 1: A man is playing a guitar. — **Hypothesis 1:** A person is making music.
Prediction: Entailment — **Ground Truth:** Entailment
Premise 2: A dog is barking loudly. — **Hypothesis 2:** Nothing is making noise.
Prediction: Contradiction — **Ground Truth:** Contradiction

In both cases, the model accurately captures key semantic relationships. It generalizes from "man" to "person" and correctly infers that "playing a guitar" entails "making music." Similarly, it identifies that a barking dog contradicts the absence of noise. These examples illustrate the model's ability to reason effectively when the premise-hypothesis relation is clear and well-supported by the training data.

4.2.2 Wrong Predictions

These two examples expose two key weaknesses in the model's reasoning: a tendency to misinterpret negation and an overreliance on surface-level lexical associations.
Premise 1: Two men are sitting in the sun. — **Hypothesis 1:** Nobody is sitting in the shade.
Premise 2: A man is walking a dog. — **Hypothesis 2:** No cat is outside.
Prediction (both): Contradiction — **Ground Truth (both):** Neutral

These cases show that the model confuses contrast with contradiction. When the hypothesis includes a word that seems opposite to something in the premise, like "shade" after "sun" or "cat" after "dog," the model assumes the sentences are in conflict. The presence of negation reinforces this assumption. But the sentences do not contradict each other; they simply mention different things. The model struggles to recognize when two statements are logically unrelated.

5 Conclusions

This practical showed that sentence encoders trained on SNLI can generalize to a range of downstream tasks. The BiLSTM-Max model performed best across both in-domain and transfer evaluations, confirming that bidirectional processing and max pooling lead to stronger representations. My results follow the trends observed in Conneau et al. [3]. Simpler models are faster but less effective, and static embeddings limit the model's ability to reason deeply. Overall, the results highlight the importance of architecture choice when learning general-purpose sentence embeddings.

References

[1] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. In *EMNLP*, 2015.

[2] Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. In *Proceedings of LREC 2018*, 2018.

[3] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017.