Revisiting the Importance of Encoding Logic Rules in Sentiment Classification

Kalpesh Krishna, Preethi Jyothi, Mohit Iyyer

Presented by Chi Zhang

Feb 25, 2019
Introduction

• Task: Sentence level sentiment classification
  • Focus on syntactically-complex inputs like $A$-but-$B$ sentences.

• Contribution
  • To meaningful compare different models, their accuracies must be averaged over far more random seeds than what has traditionally been reported.
  • Distillation model in [1] is ineffective while contextualized ELMo [2] embeddings yield significantly better performance.
  • Analyze and visualize how ELMo implicitly learn logic rules.
  • Crowdsourced analysis reveals how ELMo outperforms baseline models even on sentences with ambiguous sentiment labels.

Importance of Performance Averaging

• 100 random seeds on Stanford Sentiment Treebank (SST-2) dataset using CNN sentiment classification [1] baseline.
• The range of accuracies range from 83.47 to 87.20.
• The variance even persists after average converges.
• Conclusion: To be reproducible, only averaged accuracies should be reported in this task and dataset.

[1] Kim et al. Convolutional Neural Networks for Sentence Classification
Harness DNN with Logic Rules
Projection

• Goal: explicitly inject logic rules into neural network parameters

• Idea:
  • Given network $p_\theta(y|x)$, project it into $q_\theta(y|x)$ such that

$$\min_{q,\xi \geq 0} \text{KL}(q(Y|X) \| p_\theta(Y|X)) + C \sum_{l,g_l} \xi_{l,g_l}$$

$$\text{s.t. } \lambda_l (1 - \mathbb{E}_q [r_{l,g_l}(X,Y)]) \leq \xi_{l,g_l}$$

$$gL = 1, \ldots, GL, \ l = 1, \ldots, L$$

• Close-form solution

$$q^*(Y|X) \propto p_\theta(Y|X) \exp \left\{- \sum_{l,g_l} C \lambda_l (1 - r_{l,g_l}(X,Y)) \right\}$$

$\lambda_l$: confidence of rule $l$.

$r_{l,g_l}$: groundings of rule $l$ on sentence $X$ given label $Y$.

$\xi$: Slack variable.

A-but-B: sentiment is equal to B.

has-‘A-but-B’-structure($S$) $\Rightarrow$

$(1(y = +) \Rightarrow \sigma_\theta(B) + \land \sigma_\theta(B) + \Rightarrow 1(y = +))$, [1] Hu et al. Harnessing Deep Neural Networks with Logic Rules.
Knowledge Distillation

• Update student $p_\theta(y|x)$ using hard labels (0/1) and soft labels (logits) from teacher network $q_\theta(y|x)$
Iteratively update student and teacher

**Algorithm 1** Harnessing NN with Rules

**Input:** The training data \( D = \{(x_n, y_n)\}_{n=1}^N \),
- The rule set \( R = \{(R_l, \lambda_l)\}_{l=1}^L \),
- Parameters: \( \pi \) – imitation parameter
  \( C \) – regularization strength

1: Initialize neural network parameter \( \theta \)
2: repeat
3: Sample a minibatch \( (X, Y) \subset D \)
4: Construct teacher network \( q \) with Eq.(4)
5: Transfer knowledge into \( p_\theta \) by updating \( \theta \) with Eq.(2)
6: until convergence

**Output:** Distill student network \( p_\theta \) and teacher network \( q \)
Is Knowledge Distillation really useful?

• Why can’t we train $p$ with sentiment labels and do a final projection to obtain $q$?

• Yes, we can!

<table>
<thead>
<tr>
<th></th>
<th>Reported Test Accuracy (Hu et al., 2016)</th>
<th>Averaged Test Accuracy</th>
<th>Averaged $A$-$but$-$B$ accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>no-distill</td>
<td>distill</td>
<td>no-distill</td>
</tr>
<tr>
<td>no-project</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>87.2</td>
<td>+1.6</td>
<td>88.8</td>
</tr>
<tr>
<td></td>
<td>+0.7</td>
<td>+0.5</td>
<td></td>
</tr>
<tr>
<td>project</td>
<td>87.9</td>
<td>+1.4</td>
<td>89.3</td>
</tr>
</tbody>
</table>
Contextualized Embedding ELMo

- Contextualized Language Models
- Maximize bidirectional log likelihood.
- State-of-art NLP model on a variety tasks for 6 months
- BERT is now the state-of-art.

\[
\sum_{k=1}^{N} \left( \log p(t_k | t_1, \ldots, t_{k-1}; \Theta_x, \hat{\Theta}_{LSTM}, \Theta_s) \\
+ \log p(t_k | t_{k+1}, \ldots, t_N; \Theta_x, \hat{\Theta}_{LSTM}, \Theta_s) \right).
\]

[1] Peters et al. Deep contextualized word representations
Performance of ELMo

• KL Divergence between $q$ and $p$: 0.27, 0.26, 0.13 for [1], [2], [3].

<table>
<thead>
<tr>
<th>Model</th>
<th>Test</th>
<th>but</th>
<th>but or neg</th>
</tr>
</thead>
<tbody>
<tr>
<td>no-distill</td>
<td>85.98</td>
<td>78.69</td>
<td>80.13</td>
</tr>
<tr>
<td>no-distill</td>
<td>86.54</td>
<td>83.40</td>
<td></td>
</tr>
<tr>
<td>distill 7</td>
<td>86.11</td>
<td>79.04</td>
<td></td>
</tr>
<tr>
<td>distill</td>
<td>86.62</td>
<td>83.32</td>
<td></td>
</tr>
<tr>
<td>ELMo</td>
<td>88.89</td>
<td>86.51</td>
<td>87.24</td>
</tr>
<tr>
<td>ELMo</td>
<td>88.96</td>
<td>87.20</td>
<td></td>
</tr>
</tbody>
</table>

[1] Kim et al. [Convolutional Neural Networks for Sentence Classification](https://example.com)
Intra-sentence Similarity

• The words within the A and within the B part of the A-but-B sentence share the same part of the vector space.

Figure 3: Heat map showing the cosine similarity between pairs of word vectors within a single sentence. The left figure has fine-tuned word2vec embeddings. The middle figure contains the original ELMo embeddings without any fine-tuning. The right figure contains fine-tuned ELMo embeddings. For better visualization, the cosine similarity between identical words has been set equal to the minimum value in the heat map.
Crowdsourced Experiments

• Goal: Analyze how ambiguity impact the performance.

• *Nine* workers scored the sentiment of each *A-but-B* and negation sentence in the test SST2 split as 0 (negative), 0.5 (neutral) or 1 (positive) and average the score.

• Set threshold $x$ and label sentence score $(x, 1]$ as positive, $[0, 1 - x)$ as negative and $[1 - x, x]$ as neutral. Higher thresholds correspond to sets of less ambiguous sentences.
Crowdsourced Performance

Figure 4: Average performance on the A-but-B part of the crowd-sourced dataset (210 sentences, 100 seeds). For each threshold, only non-neutral sentences are used for evaluation.
Questions?