Self-Adaptive Hierarchical Sentence Model

Han Zhao, Zhengdong Lu and Pascal Poupart

1{han.zhao, ppoupart}@uwaterloo.ca, 2lu.zhengdong@huawei.com
1University of Waterloo, 2Noah’s Ark Lab, Huawei Technologies

Introduction

We propose a self-adaptive hierarchical sentence model (AdaSent) to represent phrases/short sentences in a hierarchy.

We apply the mixture-of-experts framework to summarize representations of different granularities to make a final consensus.

AdaSent is able to automatically learn the representation that is suitable for the task at hand through proper training.

Empirical studies on 5 benchmark data sets show the superiority of AdaSent over previous approaches.

Architecture

Structure:

For an input sentence with length \( T \), AdaSent builds a directed acyclic graph with \( T \) levels.

Word embeddings are mapped from \( \mathbb{R}^d \rightarrow \mathbb{R}^D \) at the bottom, where \( d \leq D \).

Hidden units on \( t \)-th level contain intermediate representation for phrases of length \( t \).

Unit on the top is the global representation for the sentence.

Local Composition:

\[
\begin{align*}
    h_j^t &= \omega_0 h_j^{t-1} + \omega_1 h_j^{t-1} + \omega_2 h_j^{t-1}
    &\quad + \omega_3 h_j^{t-1} + b_j
    \end{align*}
\]

Max Pooling:

\[
\bar{h}^t_i = \max_{j=1}^{T-t+1} h_j^t, \forall i \in 1:D
\]

Gating Network: A gating network takes \( \bar{h}^t \in \mathbb{R}^D, t = 1:T \) as input and outputs a belief score \( 0 \leq \gamma_i \leq 1 \) that depicts how confident the \( t \)-th level summarization in the hierarchy is suitable to be used as a proper representation of the current input instance for the task at hand. We require \( \sum_{i=1}^{D} \gamma_i = 1 \).

Decision Consensus:

\[
p(C = c|\mathbf{x}_t; T) = \sum_{t=1}^{T} p(C = c|\mathbf{h}_t) \cdot p(h_t = t|\mathbf{x}_t) = \sum_{t=1}^{T} g(h_t) \cdot w(h_t)
\]

where \( g(\cdot) \) is the classification function and \( w(\cdot) \) is the gating network.

Learning:

\[
\min \frac{1}{N} \sum_{n=1}^{N} \mathcal{L}(x_n, y_n) + \lambda \sum_{i=1}^{T} \mathcal{D}(\mathbf{W}_i^H) + \mathcal{D}(\mathbf{W}_R)
\]

where \( \mathcal{L}(\cdot, \cdot) \) is the negative class conditional log-likelihood function.

We use mini-batch AdaGrad to optimize the objective. Compute partial derivatives using back-propagation through structure:

\[
\frac{\partial L}{\partial W_L} = \frac{\partial L}{\partial h_j^{t+1}} \frac{\partial h_j^{t+1}}{\partial h_j^{t+1}} + \frac{\partial L}{\partial h_j^{t+1}} \frac{\partial h_j^{t+1}}{\partial h_j} \frac{\partial h_j}{\partial W_R}
\]

Experiments

Data Sets:

\[
\begin{array}{|c|c|c|c|c|}
\hline
\text{Data} & \text{MR} & \text{CR} & \text{SUBJ} & \text{MPQA} & \text{TREC} \\
\hline
\text{NB-SVM} & 79.4 & 81.8 & 93.2 & 86.3 & - \\
\text{MNB} & 79.0 & 80.0 & 93.6 & 86.3 & - \\
\text{RAE} & 77.7 & - & - & 86.4 & - \\
\text{MV-RecNN} & 79.0 & - & - & - & - \\
\text{CNN} & 81.5 & 85.0 & 93.4 & 89.6 & 93.6 \\
\text{DCNN} & - & - & - & 93.0 & - \\
\text{P.V.} & 74.8 & 78.1 & 90.5 & 74.2 & 91.8 \\
\text{cBoW} & 77.2 & 79.9 & 91.3 & 86.4 & 87.3 \\
\text{RNN} & 77.2 & 82.3 & 93.7 & 90.1 & 90.2 \\
\text{BRNN} & 82.3 & 82.6 & 94.2 & 90.3 & 91.0 \\
\text{GrConv} & 76.3 & 81.3 & 89.5 & 84.5 & 88.4 \\
\text{AdaSent} & 83.1 & 86.3 & 95.5 & 93.3 & 92.4 \\
\hline
\end{array}
\]

Classification Accuracy:

Representations: