TDNN: A Two-stage Deep Neural Network for Prompt-independent Automated Essay Scoring

Cancan Jin$^1$ Ben He$^{1,3}$ Kai Hui$^2$ Le Sun$^{3,4}$

$^1$School of Computer & Control Engineering, University of Chinese Academy of Sciences, Beijing, China
$^2$ SAP SE, Berlin, Germany
$^3$ Institute of Software, Chinese Academy of Sciences, Beijing, China
$^4$ Beijing Advanced Innovation Center for Language Resources, Beijing, China

jincancan15@mails.ucas.ac.cn, benhe@ucas.ac.cn
kai.hui@sap.com, sunle@iscas.ac.cn
Outline

• Background
• Method
• Experiments
• Conclusions
What is Automated Essay Scoring (AES)?

- Computer produces summative assessment for evaluation
- Aim: reduce human workload
- AES has been put into practical use by ETS from 1999
Prompt-specific and -Independent AES

• Most existing AES approaches are **prompt-specific**
  – Require human labels for each prompt to train
  – Can achieve satisfying human-machine agreement
    • Quadratic weighted kappa (QWK) > 0.75 [Taghipour & Ng, EMNLP 2016]
    • Inter-human agreement: QWK=0.754

• **Prompt-independent** AES remains a challenge
  – Only non-target human labels are available
Challenges in Prompt-independent AES

Source Prompts

- Prompt 1: Winter Olympics
- Prompt 2: Rugby World Cup
- Prompt 3: Australian Open

Learn

Model

Predict

Target Prompt

World Cup 2018
Challenges in Prompt-independent AES

Source Prompts

Prompt 1: Winter Olympics
Prompt 2: Rugby World Cup
Prompt 3: Australian Open

Target Prompt

Unavailability of rated essays written for the target prompt
Challenges in Prompt-independent AES

- Previous approaches learn on source prompts
  - Domain adaption [Phandi et al. EMNLP 2015]
  - Cross-domain learning [Dong & Zhang, EMNLP 2016]
  - Achieved Avg. QWK = 0.6395 at best with up to 100 labeled target essays
Challenges in Prompt-independent AES

Off-topic: essays written for source prompts are mostly irrelevant
Outline

• Background
• Method
• Experiments
• Conclusions
TDNN: A Two-stage Deep Neural Network for Prompt-independent AES

- Based on the idea of transductive transfer learning
- Learn on target essays
- Utilize the content of target essays to rate
The Two-stage Architecture

- Prompt-independent stage: train a shallow model to create pseudo labels on the target prompt
The Two-stage Architecture

- Prompt-dependent stage: learn an end-to-end model to predict essay ratings for the target prompts
Prompt-independent stage

• Train a robust prompt-independent AES model
  • Using Non-target prompts
  • Learning algorithm: RankSVM for AES
  • Pre-defined prompt-independent features

• Select confident essays written for the target prompt
Prompt-independent stage

- Train a robust prompt-independent AES model
  - Using Non-target prompts
  - Learning algorithm: RankSVM
  - Pre-defined prompt-independent features

- Select confident essays written for the target prompt

| Predicted Scores | 0 | 10 |
Prompt-independent stage

• Train a robust prompt-independent AES model
  • Using Non-target prompts
  • Learning algorithm: RankSVM
  • Pre-defined prompt-independent features

• Select confident essays written for the target prompt

Predicted Scores

\begin{center}
\begin{tabular}{c|c|c}
0 & 4 & 10 \\
\end{tabular}
\end{center}

Predicted ratings in $[0, 4]$ as negative examples
Prompt-independent stage

- Train a **robust** prompt-independent AES model
  - Using Non-target prompts
  - Learning algorithm: **RankSVM**
  - Pre-defined **prompt-independent features**

- Select **confident** essays written for the target prompt

| Predicted Scores | 0 | 4 | 8 | 10 |

Predicted ratings in [8, 10] as **positive** examples
Prompt-independent stage

- Train a robust prompt-independent AES model
  - Using Non-target prompts
  - Learning algorithm: RankSVM
  - Pre-defined prompt-independent features

- Select confident essays written for the target prompt

Predicted Scores

\[
\begin{array}{c c c}
0 & 4 & 8 & 10 \\
0 & 1 \\
\end{array}
\]

Converted to 0/1 labels
Prompt-independent stage

- Train a robust prompt-independent AES model
  - Using Non-target prompts
  - Learning algorithm: RankSVM
  - Pre-defined prompt-independent features

- Select confident essays written for the target prompt
  - Common sense: $\geq 8$ is good, $< 5$ is bad
  - Enlarge sample size
Prompt-dependent stage

- Train a hybrid deep model for a prompt-dependent assessment

- An end-to-end neural network with three parts of inputs:
  - Word semantic embeddings
  - Part-of-speech (POS) taggings
  - Syntactic taggings
Architecture of the hybrid deep model

- **Bi-LSTM:**
  - Semantic Network: $\vec{w}_1, \vec{w}_2, \vec{w}_3, \ldots$
  - POS Network: $\tilde{\vec{w}}_1, \tilde{\vec{w}}_2, \tilde{\vec{w}}_3, \ldots$
  - Syntactic Network: $\tilde{S}_1, \tilde{S}_2, \tilde{S}_3, \ldots$

- **Sentence Layer:**
  - Essay Layer: $\vec{e}_{POS}$
  - Essay Layer: $\vec{e}_{Synt}$

- **Phrase Layer:**
  - Essay Layer: $\vec{e}_{Sem}$

- **Feed-forward Network:**
  - Concatenate $r$

Multi-layer structure: Words – (phrases) - Sentences – Essay
Architecture of the hybrid deep model

Glove word embeddings
Architecture of the hybrid deep model

Feed-forward Network

\[ \tilde{e}_{Sem} \]
\[ \tilde{s}_1, \tilde{s}_2, \ldots \]
\[ \tilde{w}_1, \tilde{w}_2, \tilde{w}_3, \ldots \]

Semantic Network

\[ \tilde{e}_{POS} \]
\[ \tilde{s}_1, \tilde{s}_2, \ldots \]
\[ \tilde{w}_1, \tilde{w}_2, \tilde{w}_3, \ldots \]

POS Network

\[ \tilde{e}_{Synt} \]
\[ \tilde{s}_1, \tilde{s}_2, \ldots \]
\[ \tilde{w}_1, \tilde{w}_2, \tilde{w}_3, \ldots \]
\[ \tilde{t}_1, \tilde{t}_2, \tilde{t}_3, \ldots \]

Syntactic Network

\[ \tilde{e} \]
\[ \tilde{s}_1, \tilde{s}_2, \ldots \]
\[ \tilde{w}_1, \tilde{w}_2, \tilde{w}_3, \ldots \]
\[ \tilde{p}_{r_1}, \tilde{p}_{r_2}, \ldots \]

Phrase Layer

Sentence Layer

Bi-LSTM

Concatenate

Part-of-speech taggings
Architecture of the hybrid deep model

Feed-forward Network → \( r \)

\[ \tilde{e}_{Sem} \]
\[ \tilde{e}_{POS} \]
\[ \tilde{e}_{Synt} \]

Sentence Layer

Sentence Layer

Sentence Layer

Bi-LSTM

Bi-LSTM

Bi-LSTM

\[ \tilde{S}_1, \tilde{S}_2, \ldots \]

\[ \tilde{S}_1, \tilde{S}_2, \ldots \]

\[ \tilde{S}_1, \tilde{S}_2, \ldots \]

Bi-LSTM

Bi-LSTM

Bi-LSTM

\[ \tilde{w}_1, \tilde{w}_2, \tilde{w}_3, \ldots \]

\[ \tilde{w}_1, \tilde{w}_2, \tilde{w}_3, \ldots \]

\[ \tilde{w}_1, \tilde{w}_2, \tilde{w}_3, \ldots \]

Semantic Network

POS Network

Syntactic Network

\[ \tilde{p}_r_1, \tilde{p}_r_2, \ldots \]

Sentence Layer

Phrase Layer

Syntactic taggings
Architecture of the hybrid deep model

Multi-layer structure: Words – (phrases) - Sentences – Essay
Architecture of the hybrid deep model
Model Training

• Training loss: **MSE on 0/1 pseudo labels**

• Validation metric: **Kappa on 30% non-target essays**
  – Select the model that can best **rate**
Outline

- Background
- Method
- Experiments
- Conclusions
Dataset & Metrics

• We use the standard **ASAP** corpus
  – 8 prompts with >10K essays in total
• **Prompt-independent AES**: 7 prompts are used for training, 1 for testing

• Report on common human-machine agreement metrics
  – Pearson’s correlation coefficient (PCC)
  – Spearman’s correlation coefficient (SCC)
  – Quadratic weighted Kappa (QWK)
Baselines

- **RankSVM** based on prompt-independent handcrafted features
  - Also used in the prompt-independent stage in TDNN
- **2L-LSTM** [Alikaniotis et al., ACL 2016]
  - Two LSTM layer + linear layer
- **CNN-LSTM** [Taghipour & Ng, EMNLP 2016]
  - CNN + LSTM + linear layer
- **CNN-LSTM-ATT** [Dong et al., CoNLL 2017]
  - CNN-LSTM + attention
• High variance of DNN models’ performance on all 8 prompts
• Possibly caused by learning on non-target prompts
• RankSVM appears to be the most stable baseline
• Justifies the use of RankSVM in the first stage of TDNN
TDNN outperforms the best baseline on 7 out of 8 prompts.
Performance improvements gained by learning on the target prompt.
### Average performance on 8 prompts

<table>
<thead>
<tr>
<th>Method</th>
<th>QWK</th>
<th>PCC</th>
<th>SCC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Baselines</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RankSVM</td>
<td>.5462</td>
<td>.6072</td>
<td>.5976</td>
</tr>
<tr>
<td>2L-LSTM</td>
<td>.4687</td>
<td>.6548</td>
<td>.6214</td>
</tr>
<tr>
<td>CNN-LSTM</td>
<td>.5362</td>
<td>.6569</td>
<td>.6139</td>
</tr>
<tr>
<td>CNN-LSTM-ATT</td>
<td>.5057</td>
<td>.6535</td>
<td>.6368</td>
</tr>
<tr>
<td><strong>TDNN</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TDNN(Sem)</td>
<td>.5875</td>
<td>.6779</td>
<td>.6795</td>
</tr>
<tr>
<td>TDNN(Sem+POS)</td>
<td>.6582</td>
<td>.7103</td>
<td>.7130</td>
</tr>
<tr>
<td>TDNN(Sem+Synt)</td>
<td>.6856</td>
<td>.7244</td>
<td>.7365</td>
</tr>
<tr>
<td>TDNN(POS+Synt)</td>
<td>.6784</td>
<td>.7189</td>
<td>.7322</td>
</tr>
<tr>
<td>TDNN(ALL)</td>
<td>.6682</td>
<td>.7176</td>
<td>.7258</td>
</tr>
</tbody>
</table>
## Average performance on 8 prompts

<table>
<thead>
<tr>
<th>Method</th>
<th>QWK</th>
<th>PCC</th>
<th>SCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baselines</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RankSVM</td>
<td>.5462</td>
<td>.6072</td>
<td>.5976</td>
</tr>
<tr>
<td>2L-LSTM</td>
<td>.4687</td>
<td>.6548</td>
<td>.6214</td>
</tr>
<tr>
<td>CNN-LSTM</td>
<td>.5362</td>
<td>.6569</td>
<td>.6139</td>
</tr>
<tr>
<td>CNN-LSTM-ATT</td>
<td>.5057</td>
<td>.6535</td>
<td>.6368</td>
</tr>
<tr>
<td>TDNN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TDNN(Sem)</td>
<td>.5875</td>
<td>.6779</td>
<td>.6795</td>
</tr>
<tr>
<td>TDNN(Sem+POS)</td>
<td>.6582</td>
<td>.7103</td>
<td>.7130</td>
</tr>
<tr>
<td><strong>TDNN(Sem+Synt)</strong></td>
<td><strong>.6856</strong></td>
<td><strong>.7244</strong></td>
<td><strong>.7365</strong></td>
</tr>
<tr>
<td>TDNN(POS+Synt)</td>
<td>.6784</td>
<td>.7189</td>
<td>.7322</td>
</tr>
<tr>
<td>TDNN(ALL)</td>
<td>.6682</td>
<td>.7176</td>
<td>.7258</td>
</tr>
</tbody>
</table>
## Average performance on 8 prompts

<table>
<thead>
<tr>
<th>Method</th>
<th>QWK</th>
<th>PCC</th>
<th>SCC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Baselines</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RankSVM</td>
<td>.5462</td>
<td>.6072</td>
<td>.5976</td>
</tr>
<tr>
<td>2L-LSTM</td>
<td>.4687</td>
<td>.6548</td>
<td>.6214</td>
</tr>
<tr>
<td>CNN-LSTM</td>
<td>.5362</td>
<td>.6569</td>
<td>.6139</td>
</tr>
<tr>
<td>CNN-LSTM-ATT</td>
<td>.5057</td>
<td>.6535</td>
<td>.6368</td>
</tr>
<tr>
<td><strong>TDNN</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TDNN(Sem)</td>
<td>.5875</td>
<td>.6779</td>
<td>.6795</td>
</tr>
<tr>
<td>TDNN(Sem+POS)</td>
<td>.6582</td>
<td>.7103</td>
<td>.7130</td>
</tr>
<tr>
<td>TDNN(Sem+Synt)</td>
<td>.6856</td>
<td>.7244</td>
<td>.7365</td>
</tr>
<tr>
<td>TDNN(POS+Synt)</td>
<td>.6784</td>
<td>.7189</td>
<td>.7322</td>
</tr>
<tr>
<td><strong>TDNN(ALL)</strong></td>
<td>.6682</td>
<td>.7176</td>
<td>.7258</td>
</tr>
</tbody>
</table>
Sanity Check: Relative Precision

How the quality of pseudo examples affects the performance of TDNN?

➢ The sanctity of the selected essays, namely, the number of positive (negative) essays that are better (worse) than all negative (positive) essays.

➢ Such relative precision is at least 80% and mostly beyond 90% on different prompts

➢ TDNN can at least learn from correct 0/1 labels
Conclusions

• It is beneficial to learn an AES model on the target prompt
• Syntactic features are useful addition to the widely used Word2Vec embeddings
• Sanity check: small overlap between pos/neg examples
• Prompt-independent AES remains an open problem
  – ETS wants Kappa>0.70
  – TDNN can achieve 0.68 at best
Thank you!