ALICE++: Adversarial Training for Robust and Effective Temporal Reasoning

Lis Kanashiro Pereira  
Ochanomizu University  
kanashiro.pereira@ocha.ac.jp

Fei Cheng  
Kyoto University, Japan  
feicheng@i.kyoto-u.ac.jp

Masayuki Asahara  
NINJAL, Japan  
masayu-a@ninja.l.ac.jp

Ichiro Kobayashi  
Ochanomizu University, Japan  
koba@is.is.ac.jp

Abstract

We propose an enhanced adversarial training algorithm for fine-tuning transformer-based language models (i.e., RoBERTa) and apply it to the temporal reasoning task. Instead of adding the perturbation only to the embedding layer, our algorithm searches for the best combination of layers to add the adversarial perturbation. We further enhance this algorithm with $f$-divergences, i.e., the Jensen-Shannon divergence. Moreover, we enrich this model with general commonsense knowledge by leveraging data from the general commonsense knowledge task in a multi-task learning scenario. Our results show that our model can improve performance on both English and Japanese temporal reasoning benchmarks, and establishes new state-of-the-art results.

Although recent pre-trained language models such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) have achieved great success in a wide range of natural language processing (NLP) tasks, these models may still perform poorly on temporal reasoning scenarios. Ribeiro et al. (2020) has shown that such models often fail to make even simple temporal distinctions, for example, to distinguish the words before and after, resulting in degraded performance.

Following best practices from recent work on enhancing model generalization and robustness, we propose a model that effectively leverages pretrained representations (i.e. RoBERTa), adversarial training, and multi-task learning for robust temporal reasoning. More specifically, our main contributions are: 1) we propose an enhanced adversarial training algorithm for fine-tuning transformer-based language models that boosts the fine-tuning performance of RoBERTa. More specifically, our algorithm generates and adds the perturbation to a combination of layers during adversarial training. We hypothesize this might encourage the model to generate more diverse adversarial examples, and improve model generalization capability. Common adversarial training approaches for NLP add the perturbation only to the embedding layer (Zhu et al., 2019; Jiang et al., 2019; Liu et al., 2020; Pereira et al., 2020). In addition, we further enhance this algorithm with $f$-divergences (i.e., the Jensen-Shannon divergence), recently proposed by Cheng et al. (2021); 2) we enrich this model with general commonsense knowledge by leveraging data from the general commonsense knowledge task in a multi-task learning scenario; 3) we apply our model to several temporal reasoning tasks and improve state-of-the-art results.

1 Background

In this section, we describe the temporal reasoning tasks we tackle in this work. All tasks are challenging since they require deep understanding of the temporal properties of language.

Event Ordering Prediction Task: This task involves predicting the temporal relationship between a pair of input events in a span of text. We use the MATRES dataset (Ning et al., 2018). It originally contains 13,577 pairs of events annotated with a temporal relation (BEFORE, AFTER, EQUAL, VAGUE). The temporal annotations are performed on 256 English documents (and 20 more for evalua-
tion) from the TimeBank (Pustejovsky et al., 2003), AQUAINT (Graff, 2002) and Platinum (UzZaman et al., 2013) datasets. An example of a sentence with two events (in bold) that hold the BEFORE relation is below:

At one point, when it (e1:became) clear controllers could not contact the plane, someone (e2:said) a prayer.

We follow Zhou et al. (2021), and we train and evaluate only the instances with a label of either “BEFORE” or “AFTER”.

**Event Duration Prediction Task:** This task consists of deciding whether a given event has a duration longer or shorter than a day. We use TimeML (Saurí et al., 2006; Pan et al., 2006), a dataset with event duration annotated as lower and upper bounds. An example of a sentence with an event (in bold) that has a duration shorter than a day is shown below:

In Singapore, stocks hit a five year low.

**Story Cloze Task (SCT):** This task involves choosing an ending to a story. We use the Story Cloze Task dataset (Mostafazadeh et al., 2017), where the task is to choose the correct ending, among two choices, to a 4-sentence story. It captures a rich set of causal and temporal commonsense relations between daily events. An example from the dataset is below. The correct answer is in **bold**.

*Story:* Danny bought a boat. His nearby marina was having a race. He decided to enter. Danny and his best friend manned the boat.

a) Danny decided to go to sleep.

b) They prepared for the start of the race.

d) They prepared for the start of the race.

**Temporal Commonsense Reasoning Task:** This task focuses on temporal commonsense reasoning. We use the MC-TACO (Zhou et al., 2019) dataset. It considers five temporal properties: (1) duration (how long an event takes), (2) temporal ordering (typical order of events), (3) typical time (when an event occurs), (4) frequency (how often an event occurs), and (5) stationarity (whether a state is maintained for a very long time or indefinitely). It contains 13k tuples, each consisting of a sentence, a question, and a candidate answer, that should be judged as plausible or not. The sentences are taken from different sources such as news, Wikipedia, and textbooks. An example from the dataset is below. The correct answer is in **bold**.

*Paragraph:* Growing up on a farm near St. Paul, L. Mark Bailey didn’t dream of becoming a judge.

*Question:* How many years did it take for Mark to become a judge?

a) 63 years & b) 7 weeks & c) **7 years**

d) 7 seconds & e) 7 hours &

In the next section, we introduce our temporal reasoning model.

**2 Temporal Reasoning Model**

Our model uses RoBERTa (Liu et al., 2019) as the text encoder as it has obtained high performance on several natural language understanding (NLU) benchmarks. We focus on exploring adversarial training and multi-task learning, as detailed below.

**Adversarial training (ADV):** Adversarial training works as an online data augmentation method and can help improve model performance, especially in low-resource scenarios. It can also help improve model performance without increasing the model size, which is helpful in scenarios where computational resources are limited. Adversarial training has proven effective in improving model generalization and robustness in computer vision (Madry et al., 2017; Goodfellow et al., 2014) and more recently in natural language processing (NLP) (Zhu et al., 2019; Jiang et al., 2019; Cheng et al., 2019; Liu et al., 2020; Pereira et al., 2020). It works by augmenting the input with a small perturbation that maximizes the adversarial loss:

\[
\min_{\theta} \mathbb{E}_{(x,y)\sim D} \left[ \max_{\delta} I(f(x + \delta; \theta), y) \right],
\]

where the inner maximization can be solved by projected gradient descent (Madry et al., 2017). Recently, adversarial training has been successfully applied to NLP as well (Zhu et al., 2019; Jiang et al., 2019; Pereira et al., 2020). In our work, we propose...
to enhance the ALICE (Pereira et al., 2020) algorithm. ALICE combines two approaches to estimate the perturbation $\delta$: one that uses the label $y$ (Zhu et al., 2019) and another that uses the model prediction $f(x; \theta)$, i.e., a “virtual” label (Miyato et al., 2018; Jiang et al., 2019):

$$\min_{\theta} \mathbb{E}_{(x,y) \sim D}[\max_{\delta_1} l(f(x + \delta_1; \theta), y) + \alpha \max_{\delta_2} l(f(x + \delta_2; \theta), f(x; \theta))],$$

(1)

where $\delta_1$ and $\delta_2$ are two different perturbations, bounded by a general $l_p$ norm ball, estimated by a fixed $K$ steps of the gradient-based optimization approach. In our experiments, we set $p = \infty$. Effectively, the second term encourages smoothness in the input neighborhood, and $\alpha$ is a hyperparameter that controls the trade-off between standard errors and adversarial errors. ALICE has been originally proposed for the commonsense reasoning task, however, it is a general algorithm that can be applied to other tasks as well. In our work, we show its applicability to the temporal reasoning tasks described in Section 1. Moreover, we propose to further enhance this algorithm with $f$-divergences, recently proposed by Cheng et al. (2021). Specifically, we consider the posterior regularization with the Jensen-Shannon divergence (JSD) (Lin, 1991), instead of the KL-divergence, originally proposed for ALICE. JSD is a smoothed and symmetric version of the KL-Divergence. We show in our experiments that JSD outperforms the KL-divergence on the temporal tasks. In addition, we investigate which combination of layers is best for adding the perturbation during training. ALICE originally adds the perturbation only to the embedding layer. We show that adding the perturbation to a combination of the transformer’s layers instead leads to better results. We first set a maximum layer (among all RoBERTa layers, including the embedding layer) where the adversarial perturbation can be added. In each epoch, for each mini-batch selected, we first sample noise vectors $\delta_1$ and $\delta_2$ from $\mathcal{N}(0, \sigma^2I)$, with mean 0 and variation of $\sigma^2$. A layer among the embedding layer and the maximum layer previously set is randomly chosen and the model performs adversarial steps from this layer by $K$ gradient steps. The noise inputs are then constructed by adding the perturbations $\delta_1$ and $\delta_2$ to the hidden state vector of the randomly chosen layer. Specifically, the model first performs a forward pass up to the chosen layer, then the perturbations $\delta_1$ and $\delta_2$ are separately added to its hidden states, generating two different noise inputs. For example, if the second RoBERTa layer is set as the maximum layer, a layer among the embedding layer, the first, and the second layer is randomly chosen for each mini-batch selected, and adversarial training is performed from this layer. The model is then updated according to the task-specific objective for the task. The best layer combination is chosen by using a development set. We name our enhanced model ALICE++.

**Multi-task learning (MTL):** Multi-task learning is an effective training paradigm to promote model generalization ability and performance (Caruana, 1997; Liu et al., 2015; Liu et al., 2019; Ruder, 2017; Collobert et al., 2011). It works by leveraging data from many (related) tasks. We propose to enrich the training of the temporal commonsense reasoning task and Story Cloze Task by leveraging data from the general commonsense knowledge task. Since the commonsense reasoning task commonly involves reasoning about temporal events, e.g. what event(s) might happen before or after the current event, we hypothesize that those tasks might benefit from it. In our experiments, we use the CosmosQA (Huang et al., 2019) dataset. It has 35,888 questions on 21,886 distinct contexts taken from blogs of personal narratives. Each question has four answer candidates, one of which is correct. An example from this dataset is below. The correct answer is in **bold**.

**Paragraph:** Did some errands today. My prime objectives were to get textbooks, find a computer lab, find career services, get some groceries, turn in payment plan application, and find out when KEES money kicks in. I think it acts as a refund at the end of the semester at Murray, but I would be quite happy if it would work now.

**Question:** What happens after I get the refund?

**Option 1:** I can pay my bills.

**Option 2:** I can relax.

**Option 3:** I can sleep.
Option 4: None of the above choices.

We use the MT-DNN framework (Liu et al., 2019; Liu et al., 2020), which incorporates RoBERTa as the shared text encoding layer (shared across all tasks), while the top layers are task-specific. We used the pre-trained RoBERTa model to initialize the shared layers and refined them via MTL on the temporal reasoning tasks.

3 Experiments

3.1 Datasets and Evaluation Metrics

The English datasets used in our experiments are summarized in Table 1. For TimeML, we follow the train and test splits as in (Zhou et al., 2020). For MCTACO, we follow Zhou et al (2019). For the MATRES dataset, we follow Ning et al. (2018). For the Story Cloze Task, we use the 2016 and 2018 data releases after removing duplicates. We set 20% of the TimeML, MATRES, and Story Cloze Task training data as the development set to tune the hyperparameters. For the MC-TACO dataset, no training set is available. Following Zhou et al (2019), we use the dev set for fine-tuning the model. We use 20% of this data for fine-tuning the parameters.

We evaluate the performance on MATRES in terms of accuracy and F1-score, and TimeML and Story Cloze Task in terms of accuracy. For the MC-TACO dataset, we report the exact match (EM) and F1 scores, following Zhou et al (2019). EM measures how many questions a system correctly labeled all candidate answers, while F1 measures the average overlap between one’s predictions and the ground truth.

3.2 Implementation Details

Our model implementation is based on the MT-DNN framework (Liu et al., 2019; Liu et al., 2020). We use RoBERTa LARGE (Liu et al., 2019) as the text encoder. We used ADAM (Kingma and Ba, 2014) as our optimizer with a learning rate in the range \( \in \{9 \times 10^{-6}, 1 \times 10^{-5}\} \) and a batch size in the range \( \in \{16, 32, 64\} \). The maximum number of epochs was set to 10. A linear learning rate decay schedule with warm-up over 0.1 was used unless stated otherwise. To avoid gradient exploding, we clipped the gradient norm within 1. All the texts were tokenized using WordPiece and were chopped to spans no longer than 512 tokens. We also set the dropout rate of all the task-specific layers as 0.3. During adversarial training, we follow (Jiang et al., 2019) and set the perturbation size to \( 1 \times 10^{-5} \), the step size to \( 1 \times 10^{-3} \), and to \( 1 \times 10^{-5} \) the variance for initializing perturbation. We search the regularization weight \( \alpha \) in \{0.01, 0.1, 1\}. We set the number of projected gradient steps to 1.

3.3 Main Results

We present our results in Table 2. We compare our model, ALICE++, with other state-of-the-art models. Overall, the adversarial methods, i.e., ALICE and ALICE++, were able to outperform the standard fine-tuning approach (STD) and the other baselines, without using any additional knowledge source, and without using any additional dataset other than the target task datasets. These results suggest that adversarial training leads to a more robust model and helps generalize better on unseen data.

Both ALICE++ (JSD), the model that uses the Jensen-Shannon Divergence, and ALICE++ (JSD + Best layers selection), the model that uses JSD and the best layer combination to add the perturbation, were able to outperform ALICE and the other baselines. Overall, ALICE++ (JSD + Best layers selection) obtained better performance. This indicates that adding the adversarial perturbation to the other layers of the model in addition to the embedding layer can improve the model generalization capability.

For example, on the MATRES dataset, ALICE++ (JSD + Best layers selection) obtained a 89.82% F1-score, a 2.52% improvement over SYMTIME (Zhou et al., 2021), a T5 model that exploits distant supervision signals from large-scale text and uses temporal rules to combine start times and durations to infer end times. On the TimeML dataset, ALICE++ (JSD + Best layers selection) outperformed TacoML (Zhou et al., 2020), a BERT model pre-trained on explicit and implicit mentions of temporal commonsense, extracted from a large corpus using pattern rules, and obtained an accuracy of 84.45%, an absolute gain of 2.75%. On the MC-TACO dataset, ALICE++ (JSD + Best layers selection) outperforms the T5-3B model (Kaddari et al., 2020) in terms of F1-score, obtaining an F1-score of 80.09%, an improvement of 0.63%, and an EM score of 58.56%, only 0.52% lower than T5-3B.
## Table 1: Summary of the four English evaluation datasets: MATRES, TimeML, Story Cloze Task (SCT), and MC-TACO.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Train</th>
<th>#Test</th>
<th>#Label</th>
<th>Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATRES</td>
<td>10,906</td>
<td>698</td>
<td>2</td>
<td>{BEFORE, AFTER}</td>
</tr>
<tr>
<td>TimeML</td>
<td>1,248</td>
<td>1,003</td>
<td>2</td>
<td>Accuracy</td>
</tr>
<tr>
<td>SCT</td>
<td>1,571</td>
<td>1,871</td>
<td>2</td>
<td>Accuracy</td>
</tr>
<tr>
<td>MC-TACO</td>
<td>3,783</td>
<td>9,442</td>
<td>2</td>
<td>F1-Score &amp; Exact Match (EM)</td>
</tr>
</tbody>
</table>

Table 2: Test results of MATRES, TimeML, Story Cloze Task (SCT), and MC-TACO. The best results are in **bold**.

<table>
<thead>
<tr>
<th>Model</th>
<th>MATRES</th>
<th>TimeML</th>
<th>MC-TACO</th>
<th>SCT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human</td>
<td>-</td>
<td>-</td>
<td>87.70</td>
<td>75.80</td>
</tr>
<tr>
<td>STD</td>
<td>91.12</td>
<td>88.93</td>
<td>51.05</td>
<td>76.85</td>
</tr>
<tr>
<td>ALICE (Pereira et al., 2020)</td>
<td>91.69</td>
<td>91.10</td>
<td>56.45</td>
<td>79.50</td>
</tr>
<tr>
<td>ALICE++ (JSD)</td>
<td>91.55</td>
<td>89.37</td>
<td>58.10</td>
<td><strong>80.20</strong></td>
</tr>
<tr>
<td>ALICE++ (JSD + Best layers selection)</td>
<td><strong>91.98</strong></td>
<td><strong>89.82</strong></td>
<td><strong>84.45</strong></td>
<td><strong>85.56</strong></td>
</tr>
<tr>
<td>ALICE++ (JSD + Best layers selection, MT.CosmosQA)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td><strong>59.90</strong></td>
</tr>
<tr>
<td>T5-3B (Kaddari et al., 2020)</td>
<td>-</td>
<td>-</td>
<td>59.08</td>
<td>79.46</td>
</tr>
<tr>
<td>TacoML (Zhou et al., 2020)</td>
<td>-</td>
<td>81.70</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SYMTIME (Zhou et al., 2021)</td>
<td>-</td>
<td>87.30</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GDIN (Tian et al., 2020)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

# 3.4 Evaluation on Japanese dataset

We also explore the feasibility of our model on a Japanese dataset. Table 4 describes our results on the Japanese event ordering prediction task. We use the BCCWJ-Timebank corpus (Asahara et al., 2014). It consists of four tasks: 1) DCT, which denotes relations between a time expression of document creation time (DCT) and an event instance; 2) T2E, which denotes relations between a time expression (non-DCT) and an event instance within one sentence; 3) E2E, which denotes relations between two consecutive event instances; and 4) MAT, which denotes relations between two consecutive matrix verbs of event instances. We perform the document-level 5-fold cross-validation. In each split, we randomly select 15% documents as the development set. We follow a merged 6-relation set (`BE-
FORE’, ‘BEFOREOR-OVERLAP’, ‘OVERLAP’, ‘OVERLAP-ORAFTER’, ‘AFTER’, and ‘VAGUE’) as in Yoshikawa et al. (2014). The statistics of the corpus are shown in Table 3. An example from the corpus on the E2E task is shown below.

**Task: E2E**

塩少々を(e1:ふっ)てしばらく(e2:おき)、水分をふく。

(e1:Shake) the salt a little and (e2:leave) it for a while to wipe off the water.

**Label: BEFORE**

Moreover, we train all tasks jointly using multi-task learning, following Cheng et al. (2020). We use a Japanese BERT_BASE model as the text encoder. Compared to standard fine-tuning and the other baselines, ALICE++ could improve on all tasks. It outperformed the model by Cheng et al. (2020), a BERT_BASE model that dynamically updates event representations. ALICE++ also outperformed the model by Yoshikawa et al. (2014), a feature-based SVM classifier.

<table>
<thead>
<tr>
<th>Model</th>
<th>DCT</th>
<th>E2T</th>
<th>E2E</th>
<th>MAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>STD</td>
<td>83.04</td>
<td>65.15</td>
<td>68.54</td>
<td>63.50</td>
</tr>
<tr>
<td>Yoshikawa et al. (2014)</td>
<td>75.60</td>
<td>55.70</td>
<td>59.90</td>
<td>50.00</td>
</tr>
<tr>
<td>Cheng et al. (2020)</td>
<td>81.60</td>
<td>60.70</td>
<td>64.50</td>
<td>64.60</td>
</tr>
<tr>
<td>ALICE++</td>
<td>83.22</td>
<td>66.61</td>
<td>68.96</td>
<td>64.63</td>
</tr>
</tbody>
</table>

**Table 3**: Number of TLINKs in the BCCWJ-Timebank dataset. A ⟨TLINK⟩ defines the temporal ordering of temporal information expressions and event expressions.

**Table 4**: Accuracy test results on the BCCWJ-Timebank dataset. ALICE++ denotes the model that uses JSD and the best layer combination to add the perturbation.

### 4 Analysis of RoBERTa layers when adding the adversarial perturbation

In this Section, we show a brief analysis of the best combination of layers for adding the adversarial perturbation. Figure 1 shows the accuracy on the TimeML, Story Cloze Task, MC-TACO, and MALTRES development sets as we change the layer combination to add the adversarial perturbation. We can observe that adding the adversarial perturbation to the other layers of the model in addition to the embedding layer leads to better performance compared to adding the perturbation to the embedding layer only.

A similar tendency is observed on the BCCWJ-Timebank, as shown in Figure 2.

### 5 Conclusion

We proposed an adversarial training algorithm for fine-tuning transformer-based language models, ALICE++, that boosts the fine-tuning performance of RoBERTa. Our experiments demonstrated that it achieves state-of-the-art results on several temporal reasoning tasks. Although in this paper we focused on the temporal reasoning task, ALICE++ can be generalized to solve other downstream tasks as well, and we will explore this direction as to future work.
(a) Accuracy on the **TimeML** development set as we change the layer combination to add the adversarial perturbation.

(b) Accuracy on the **Story Cloze Task** development set as we change the layer combination to add the adversarial perturbation.

(c) F1-score on the **MC-TACO** development set as we change the layer combination to add the adversarial perturbation.

(d) F1-score on the **MATRES** development set as we change the layer combination to add the adversarial perturbation.

Figure 1: Performance on the TimeML, Story Cloze Task, MC-TACO, and MATRES development sets as we change the layer combination to add the adversarial perturbation. *max_layer = 0* denotes that the adversarial perturbation is added to the embedding layer only. All the other values denote that, for each mini-batch, a layer among the embedding layer and *max_layer* is randomly chosen and the model performs adversarial training from this layer. The model is then updated according to the task-specific objective for the task.
(a) Accuracy on the BCCWJ-Timebank **DCT** task development set as we change the layer combination to add the adversarial perturbation.

(b) Accuracy on the BCCWJ-Timebank **T2E** task development set as we change the layer combination to add the adversarial perturbation.

(c) Accuracy on the BCCWJ-Timebank **E2E** task development set as we change the layer combination to add the adversarial perturbation.

(d) Accuracy on the BCCWJ-Timebank **MAT** task development set as we change the layer combination to add the adversarial perturbation.

Figure 2: Accuracy on the BCCWJ-Timebank development sets as we change the layer combination to add the adversarial perturbation. *max_layer = 0* denotes that the adversarial perturbation is added to the embedding layer only. All the other values denote that, for each mini-batch, a layer among the embedding layer and *max_layer* is randomly chosen and the model performs adversarial training from this layer. The model is then updated according to the task-specific objective for the task.
Acknowledgments

This work has been supported by the project KAKENHI ID: 18H05521 and by project KAKENHI ID: 21K17802.

References


Kaddari, Z. and Mellah, Y. and Berrichtouf, T. and Belkasmi, M. G. 2020. Applying the T5 language model and duration units normalization to address temporal common sense understanding on the MCTACO dataset. In 2020 International Conference on Intelligent Systems and Computer Vision (ISCV) (pp. 1-4). IEEE.


