“Going on a vacation” takes longer than “Going for a walk”:
A Study of Temporal Commonsense Understanding

Ben Zhou,1 Daniel Khashabi,2 Qiang Ning,3 Dan Roth1
1University of Pennsylvania, 2Allen Institute for AI, 3University of Illinois at Urbana-Champaign
{xyzhou,danroth}@cis.upenn.edu danielk@allenai.org qning2@illinois.edu

Abstract

Understanding time is crucial for understanding events expressed in natural language. Because people rarely say the obvious, it is often necessary to have commonsense knowledge about various temporal aspects of events, such as duration, frequency, and temporal order. However, this important problem has so far received limited attention. This paper systematically studies this temporal commonsense problem. Specifically, we define five classes of temporal commonsense, and use crowdsourcing to develop a new dataset, MC-TACO, that serves as a test set for this task. We find that the best current methods used on MC-TACO are still far behind human performance, by about 20%, and discuss several directions for improvement. We hope that the new dataset and our study here can foster more future research on this topic.1

1 Introduction

Natural language understanding requires the ability to reason with commonsense knowledge (Schubert, 2002; Davis, 2014), and the last few years have seen significant amount of work in this direction (e.g., Zhang et al. (2017); Bauer et al. (2018); Tandon et al. (2018)). This work studies a specific type of commonsense: temporal commonsense. For instance, given two events “going on a vacation” and “going for a walk,” most humans would know that a vacation is typically longer and occurs less often than a walk, but it is still challenging for computers to understand and reason about temporal commonsense.

Note that a question may have multiple correct answers.

Figure 1: Five types of temporal commonsense in MC-TACO. Note that a question may have multiple correct answers.

Temporal commonsense has received limited attention so far. Our first contribution is that, to the best of our knowledge, we are the first to systematically study and quantify performance on a range of temporal commonsense phenomena. Specifically, we consider five temporal properties: duration (how long an event takes), temporal ordering (typical order of events), typical time (when an event happens), frequency (how often an event occurs), and stationarity (whether a state holds for a very long time or indefinitely). Previous work has investigated some of these aspects, either explicitly or implicitly (e.g., duration (Gusev et al., 2011; Williams, 2012) and ordering (Chklovski and Pantel, 2004; Ning et al., 2018b)), but none of them have defined or studied...
all aspects of temporal commonsense in a unified framework. Kozareva and Hovy (2011) defined a few temporal aspects to be investigated, but failed to quantify performances on these phenomena. Given the lack of evaluation standards and datasets for temporal commonsense, our second contribution is the development of a new dataset dedicated for it, MCTACO (short for multiple choice temporal common-sense). MCTACO is constructed via crowdsourcing with guidelines designed meticulously to guarantee its quality. When evaluated on MCTACO, a system receives a sentence providing context information, a question designed to require temporal commonsense knowledge, and multiple candidate answers (see Fig. 1; note that in our setup, more than one candidate answer can be plausible). We design the task as a binary classification: determining whether a candidate answer is plausible according to human commonsense, since there is no absolute truth here. This is aligned with other efforts that have posed commonsense as the choice of plausible alternatives (Roemmele et al., 2011). The high quality of the resulting dataset (shown in §4) also makes us believe that the notion of plausibility here is robust.

Our third contribution is that, using MCTACO as a testbed, we study the temporal commonsense understanding of the best existing NLP techniques, including ESIM (Chen et al., 2017), BERT (Devlin et al., 2019) and their variants. Results in §4 show that, despite a significant improvement over random-guess baselines, the best existing techniques are still far behind human performance on temporal commonsense understanding, indicating the need for further research in order to improve the currently limited capability to capture temporal semantics.

2 Related Work

Commonsense has been a very popular topic in recent years and existing NLP works have mainly investigated the acquisition and evaluation of commonsense in the physical world, including but not limited to, size, weight, and strength (Forbes and Choi, 2017), roundness and deliciousness (Yang et al., 2018), and intensity (Cocos et al., 2018). In terms of “events” commonsense, Rashkin et al. (2018) investigated the intent and reaction of participants of an event, and Zellers et al. (2018) tried to select the most likely subsequent event. To the best of our knowledge, no earlier work has focused on temporal commonsense, although it is critical for event understanding. For instance, Ning et al. (2018c) argues that resolving ambiguous and implicit mentions of event durations in text (a specific kind of temporal commonsense) is necessary to construct the timeline of a story.

There have also been many works trying to understand time in natural language but not necessarily the commonsense understanding of time. Most recent works include the extraction and normalization of temporal expressions (Strötgen and Gertz, 2010; Lee et al., 2014), temporal relation extraction (Ning et al., 2017, 2018d), and timeline construction (Leeuwenberg and Moens, 2018). Among these, some works are implicitly on temporal commonsense, such as event durations (Williams, 2012; Vempala et al., 2018), typical temporal ordering (Chkovski and Pantel, 2004; Ning et al., 2018a,b), and script learning (i.e., what happens next after certain events) (Granroth-Wilding and Clark, 2016; Li et al., 2018). However, existing works have not studied all five types of temporal commonsense in a unified framework as we do here, nor have they developed datasets for it.

Instead of working on each individual aspect of temporal commonsense, we formulate the problem as a machine reading comprehension task in the format of selecting plausible responses with respect to natural language queries. This relates our work to a large body of work on question-answering, an area that has seen significant progress in the past few years (Clark et al., 2018; Ostermann et al., 2018; Merkhofer et al., 2018). This area, however, has mainly focused on general natural language comprehension tasks, while we tailor it to test a specific reasoning capability, which is temporal commonsense.

<table>
<thead>
<tr>
<th>Measure</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td># of unique questions</td>
<td>1893</td>
</tr>
<tr>
<td># of unique question-answer pairs</td>
<td>13,225</td>
</tr>
<tr>
<td>avg. sentence length</td>
<td>17.8</td>
</tr>
<tr>
<td>avg. question length</td>
<td>8.2</td>
</tr>
<tr>
<td>avg. answer length</td>
<td>3.3</td>
</tr>
</tbody>
</table>

Table 1: Statistics of MCTACO.
3 Construction of MCTACO

MCTACO is comprised of $13k$ tuples, in the form of $(sentence, question, candidate answer)$; please see examples in Fig. 1 for the five phenomena studied here and Table 1 for basic statistics of it. The sentences in those tuples are randomly selected from MultiRC (Khashabi et al., 2018) (from each of its 9 domains). For each sentence, we use crowdsourcing on Amazon Mechanical Turk to collect questions and candidate answers (both correct and wrong ones). To ensure the quality of the results, we limit the annotations to native speakers and use qualification tryouts.

Step 1: Question generation. We first ask crowdsourcers to generate questions, given a sentence. To produce questions that need temporal commonsense to answer, we require that a valid question: (a) should ask about one of the five temporal phenomena we defined earlier, and (b) should not be solved simply by a word or phrase from the original sentence. We also require crowdsourcers to provide a correct answer for each of their questions, which on one hand gives us a positive candidate answer, and on the other hand ensures that the questions are answerable at least by themselves.

Step 2: Question verification. We further ask another two crowdsourcers to check the questions generated in Step 1, i.e., (a) whether the two requirements are satisfied and (b) whether the question is grammatically and logically correct. We retain only the questions where the two annotators unanimously agree with each other and the decision generated in Step 1. For valid questions, we continue to ask crowdsourcers to give one correct answer and one incorrect answer, which we treat as a seed set to automatically generate new candidate answers in the next step.

Step 3: Candidate answer expansion. Until this stage, we have collected a small set of candidate answers (3 positive and 2 negative) for each question. We automatically expand this set in three ways. First, we use a set of rules to extract numbers and quantities (“2”, “once”) and temporal terms (e.g. “a.m.”, “1990”, “afternoon”, “day”), and then randomly perturb them based on a list of temporal units (“second”), adjectives (“early”), points (“a.m.”) and adverbs (“always”). Examples are “2 a.m.” → “3 p.m.”, “1 day” → “10 days”, “once a week” → “twice a month” (more details in the appendix).

Second, we mask each individual token in a candidate answer (one at a time) and use BERT (Devlin et al., 2019) to predict replacements for each missing term; we rank those predictions by the confidence level of BERT and keep the top three.

Third, for those candidates that represent events, the previously-mentioned token-level perturbations rarely lead to interesting and diverse set of candidate answers. Furthermore, it may lead to invalid phrases (e.g., “he left the house” → “he walked the house”.) Therefore, to perturb such candidates, we create a pool of $60k$ event phrases using PropBank (Kingsbury and Palmer, 2002), and perturb the candidate answers to be the most similar ones extracted by an information retrieval (IR) system. This not only guarantees that all candidates are properly phrased, it also leads to more diverse perturbations.

We apply the above three techniques on non-“event” candidates sequentially, in the order they were explained, to expand the candidate answer set to 20 candidates per question. A perturbation technique is used, as long as the pool of candidates is still less than 20. Note there are both correct and incorrect answers in those candidates.

Step 4: Answer labeling. In this step, each $(sentence, question, answer)$ tuple produced earlier is labeled by 4 crowdsourcers, with three options: “likely”, “unlikely”, or “invalid” (sanity check for valid tuples). Different annotators may have different interpretations, yet we ensure label validity through high agreement. A tuple is kept only if all 4 annotators agree on “likely” or “unlikely”. The final statistics of MCTACO is in Table 1.

4 Experiments

We assess the quality of our dataset through human annotation, and evaluate a couple of baseline systems. We create a uniform split of $30%/70%$ of the data to dev/test. The rationale behind this split is that, a successful system has to bring in a
huge amount of world knowledge and derive commonsense understandings prior to the current task evaluation. We therefore believe that it is not reasonable to expect a system to be trained solely on this data, and we think of the development data as only providing a definition of the task. Indeed, the gains from our development data are marginal after a certain number of training instances. This intuition is studied and verified in Appendix A.2.

Evaluation metrics. Two question-level metrics are adopted in this work: exact match (EM) and F1. For a given candidate answer a that belongs to a question q, let f(a; q) ∈ {0, 1} denote the correctness of the prediction made by a fixed system (1 for correct; 0 otherwise). Additionally, let D denote the collection of questions in our evaluation set.

\[
EM \triangleq \sum_{q \in D} \prod_{a \in q} f(a; q)
\]

The recall for each question q is:

\[
R(q) = \frac{\sum_{a \in q} [f(a; q) = 1 \land \{a\} \text{is “likely”}]}{|\{a\} \text{is “likely”} \land a \in q|}
\]

Similarly, P(q) and F1(q) are defined. The aggregate F1 (across the dataset D) is the macro average of question-level F1’s:

\[
F1 \triangleq \frac{\sum_{q \in D} F1(q)}{|\{q \in D\}|}
\]

EM measures how many questions a system is able to correctly label all candidate answers, while F1 is more relaxed and measures the average overlap between one’s predictions and the ground truth.

Human performance. An expert annotator also worked on McTACO to gain a better understanding of the human performance on it. The expert answered 100 questions (about 700 (sentence, question, answer) tuples) randomly sampled from the test set, and could only see a single answer at a time, with its corresponding question and sentence.

Systems. We use two state-of-the-art systems in machine reading comprehension for this task: ESIM (Chen et al., 2017) and BERT (Devlin et al., 2019). ESIM is an effective neural model on natural language inference. We initialize the word embeddings in ESIM via either GloVe (Pennington et al., 2014) or ELMo (Peters et al., 2018) to demonstrate the effect of pre-training. BERT is a state-of-the-art contextualized representation used for a broad range of tasks. We also add unit normalization to BERT, which extracts and converts temporal expressions in candidate answers to their most proper units. For example, “30 months” will be converted to “2.5 years”. To the best of our knowledge, there are no other available systems for the “stationarity”, “typical time”, and “frequency” phenomena studied here. As for “duration” and “temporal order”, there are existing systems (e.g., Vempala et al. (2018); Ning et al. (2018b)), but they cannot be directly applied to the setting in McTACO where the inputs are natural languages.

Experimental setting. In both ESIM baselines, we model the process as a sentence-pair classification task, following the SNLI setting in AllenNLP.5 In both versions of BERT, we use the same sequence pair classification model and the same parameters as in BERT’s GLUE experiments.6 A system receives two elements at a time: (a) the concatenation of the sentence and question, and (b) the answer. The system makes a binary prediction on each instance, “likely” or “unlikely”.

Results and discussion. Table 2 compares native baselines, ESIM, BERT and their variants on the entire test set of McTACO; it also shows human performance on the subset of 100 questions.7 The system performances reported are based on default random seeds, and we observe a maximum

<table>
<thead>
<tr>
<th>System</th>
<th>F1</th>
<th>EM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>36.2</td>
<td>8.1</td>
</tr>
<tr>
<td>Always Positive</td>
<td>49.8</td>
<td>12.1</td>
</tr>
<tr>
<td>Always Negative</td>
<td>17.4</td>
<td>17.4</td>
</tr>
<tr>
<td>ESIM + GloVe</td>
<td>50.3</td>
<td>20.9</td>
</tr>
<tr>
<td>ESIM + ELMo</td>
<td>54.9</td>
<td>26.4</td>
</tr>
<tr>
<td>BERT</td>
<td>66.1</td>
<td>39.6</td>
</tr>
<tr>
<td>BERT + unit normalization</td>
<td>69.9</td>
<td>42.7</td>
</tr>
<tr>
<td>Human</td>
<td>87.1</td>
<td>75.8</td>
</tr>
</tbody>
</table>

Table 2: Summary of the performances for different baselines. All numbers are in percentages.

5https://github.com/allenai/allennlp
6https://github.com/huggingface/pytorch-pretrained-BERT
7BERT + unit normalization scored F1 = 72, EM = 45 on this subset, which is only slightly different from the corresponding number on the entire test set.
standard error of 0.8 from 3 runs on different seeds across all entries. We can confirm the good quality of this dataset based on the high performance of human annotators. ELMo and BERT improve naive baselines by a large margin, indicating that a notable amount of commonsense knowledge has been acquired via pre-training. However, even BERT still falls far behind human performance, indicating the need of further research.

We know that BERT, as a language model, is good at associating surface forms (e.g. associating “sunrise” with “morning” since they often co-occur), but may be brittle with respect to variability of temporal mentions.

Consider the following example (the correct answers are indicated with ✔ and BERT selections are underlined.) This is an example of BERT correctly associating a given event with “minute” or “hour”; however, it fails to distinguish between “1 hour” (a “likely” candidate) and “9 hours” (an “unlikely” candidate).

P: Ratner’s chairman, Gerald Ratner, said the deal remains of “substantial benefit to Ratners.”
Q: How long did the chairman speak?
✔ (a) 30 minutes ✔ (b) 1 hour
x (c) 9 hours x (d) twenty seconds

This shows that BERT does not infer a range of true answers; it instead associates discrete terms and decides individual options, which may not be the best way to handle temporal units that involve numerical values.

BERT + unit normalization is used to address this issue, but results show that it is still poor compared to human. This indicates that the information acquired by BERT is still far from solving temporal commonsense.

Since exact match (EM) is a stricter metric, it is consistently lower than F1 in Table 2. For an ideal system, the gap between EM and F1 should be small (humans only drop 11.3%). However, all other systems drop by almost 30% from F1 to EM, possibly another piece of evidence that they only associate surface forms instead of using one representation for temporal commonsense to classify all candidates.

A curious reader might ask why the human performance on this task as shown in Table 2 is not 100%. This is expected because commonsense is what most people agree on, so any single human could disagree with the gold labels in McTACO. Therefore, we think the human performance in Table 2 from a single evaluator actually indicates the good quality of McTACO.

The performance of BERT+unit normalization is not uniform across different categories (Fig. 2), which could be due to the different nature or quality of data for those temporal phenomena. For example, as shown in Table 1, “stationarity” questions have much fewer candidates and a higher random baseline.

5 Conclusion

This work has focused on temporal commonsense. We define five categories of questions that require temporal commonsense and develop a novel crowdsourcing scheme to generate McTACO, a high-quality dataset for this task. We use McTACO to probe the capability of systems on temporal commonsense understanding. We find that systems equipped with state-of-the-art language models such as ELMo and BERT are still far behind humans, thus motivating future research in this area. Our analysis sheds light on the capabilities as well as limitations of current models. We hope that this study will inspire further research on temporal commonsense.

Acknowledgements

This research is supported by a grant from the Allen Institute for Artificial Intelligence (allenai.org) and by contract HR0011-18-2-0052 and HR0011-15-C-0113 with the US Defense Advanced Research Projects Agency (DARPA). Approved for Public Release, Distribution Unlimited. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.
References


Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In NAACL.


Zhongyang Li, Xiaod Ding, and Ting Liu. 2018. Constructing narrative event evolutionary graph for script event prediction. Proc. of the International Joint Conference on Artificial Intelligence (IJCAI).


