Temporal Reasoning on Implicit Events from Distant Supervision

Ben Zhou[1,2] **Kyle Richardson**[1] **Qiang Ning**[3] **Tushar Khot**[1] **Ashish Sabharwal**[1] **Dan Roth**[2]
*[1]Allen Institute for AI [2]University of Pennsylvania [3]Amazon
{kyler,tushark,ashishs}@allenai.org {xyzhou,danroth}@cis.upenn.edu qning@amazon.com

Abstract

Existing works on temporal reasoning among events described in text focus on modeling relationships between explicitly mentioned events and do not handle event end time effectively. However, human readers can infer from natural language text many implicit events that help them better understand the situation and, consequently, better reason about time. This work proposes a new crowd-sourced dataset, TRACIE, which evaluates systems' understanding of *implicit* events – events that are not mentioned explicitly in the text but can be inferred from it. This is done via textual entailment instances querying both start and end times of events. We show that TRACIE is challenging for state-of-the-art language models. Our proposed model, SymTime, exploits distant supervision signals from the text itself and reasons over events' start time and duration to infer events' end time points. We show that our approach improves over baseline language models, gaining 5% on the i.i.d. split and 9% on an out-of-distribution test split. Our approach is also general to other annotation schemes, gaining 2%-8% on MATRES, an extrinsic temporal relation benchmark.

1 Introduction

Understanding the temporal relations between events in the narrative text is a crucial part of text understanding. When reading a story, a human can construct a latent timeline about events' start and end times, similar to the one shown in Fig. 1 about an automobile accident. This timeline not only contains the placements of explicitly mentioned events (e.g., *ride a bicycle*), but also accounts for implicit events (e.g., *distracted*). Such a latent timeline explains the dynamics between events; for example, the possible chain of events between *ride* and *recovered* in this context contains *get hit* and *injured*. The ability to construct such a timeline is essential for understanding the causal dynamics of a situation. NLP systems cannot truly understand situations and solve tasks such as scheduling, question-answering, and causal inference without being able to build such a timeline.

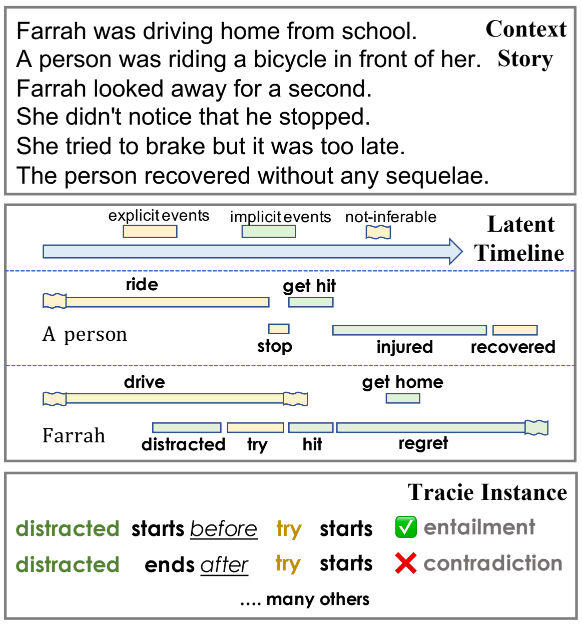

Context Story
Farrah was driving home from school.
A person was riding a bicycle in front of her.
Farrah looked away for a second.
She didn't notice that he stopped.
She tried to brake but it was too late.
The person recovered without any sequelae.
Latent Timeline
explicit events
implicit events
not-inferable
ride
get hit
A person
stop
injured
recovered
drive
get home
Farrah
distracted
try
hit
regret
Tracie Instance
distracted starts before try starts ✅ entailment
distracted ends after try starts ❌ contradiction
.... many others

Figure 1: A story, its latent timeline, and example TRACIE instances based on it.

To better evaluate this ability, we introduce a new dataset called TRACIE (*TempoRAl Closure InfErence*) that focuses on temporal relations on implicit events in short commonsense stories (Mostafazadeh et al., 2016). In addition to start times, our dataset contains high-quality annotations of end-time queries that test a system's understanding of the full temporal closure (i.e., both start and end time) of events. To avoid large-scale supervised training pitfalls, we follow Zhou et al. (2019) in minimizing the training set to a small number of samples, therefore making our dataset mainly an

* Work was done when the first and third author were affiliated with the Allen Institute for AI.

evaluation set. The final TRACIE dataset contains a total of 5.5k human-curated instances, provided in a (multi-premise) textual entailment (TE) format, as illustrated on the bottom of Fig 1.

To demonstrate the difficulty of our task, we show that existing models do not generalize well on our dataset in various settings. In a standard setting,[1] pre-trained language models such as T5-large (Raffel et al., 2020) fine-tuned on TRACIE achieves a modest binary prediction accuracy of 71.2%,[2] whereas other standard TE baselines (e.g., biLSTM encoder at ∼58%) perform near-random chance. In a more challenging *story-wide* exact match setting, where models must correctly predict all temporal relations across each story, T5 achieves a much lower accuracy of 18.6%. Consistent with other studies on temporal reasoning (Zhou et al., 2020), these results reveal serious limitations in existing pre-trained language models.

To build models more capable of understanding time with minimal direct training data, we propose a novel distant supervision technique that improves generalization by extracting temporal patterns in large-scale free text for additional model pre-training. In contrast to other attempts at extracting temporal data through patterns on sentence-level (Gusev et al., 2011; Zhou et al., 2020), our approach extracts over large windows of text such as paragraphs, in order to capture global information related to multiple events and extract signals that do not appear in small-window local contexts. By using models trained through such distant supervision in place of existing language models, we can achieve 76.1% accuracy on TRACIE (a 5% performance gain over using standard T5-Large). We also show the applicability of our technique on a standard temporal reasoning benchmark, MATRES (Ning et al., 2018b), by improving on *before* and *after* classification (from 86% to 87.5%).

We achieve further improvements by coupling our new pre-trained model with a duration model from Zhou et al. (2020) to create a new symbolic reasoning model called *SymTime*. The key idea in SymTime is to *decompose* the computation of temporal relations to the localized start-time and duration predictions, which is similarly discussed in Leeuwenberg and Moens (2018). For example, in deciding whether the *distracted* event ends before the *try* event in Fig. 1, we can use separate models to predict the start and duration of *distracted* and compare them against the start prediction of *try*. This allows better prediction on the end time, which rarely appears in the natural text and is difficult to be annotated (Ning et al., 2018b). Such a symbolic computation incorporates a logic combination over the individual models that formalizes part of the Allen interval algebra (Allen, 1983). This model, which supports a wider range of temporal computation and can be used without task-specific supervision, achieves a final accuracy of 77.5% on TRACIE's binary classification metric and a 10% improvement on the story-wide exact match.

[1]In both the training and test sets, instances are independent and identically distributed (i.i.d.).

[2]The same model achieves 77.4% on MATRES (Ning et al., 2018b) with the same amount of training instances, and 87.5% with the full training set.

2 TRACIE

In this section, we introduce the TRACIE task and dataset, including its construction and format (§2.1) and the different training/testing splits (§2.2).[3]

2.1 Task Overview and Dataset Construction

The goal of TRACIE is to test a system's ability to compare start and end times of non-extractive implicit event phrases instead of extractive trigger verbs from the context. Such tests in TRACIE take the form of textual entailment (TE) queries, similar to the multi-premise format in Lai et al. (2017). Each TRACIE instance contains 1) a **context story** (or premise) consisting of a sequence of *explicit* narrative events; 2) an **implicit event** in the form of a natural language phrase that is unmentioned but has some role in the story 3) a **comparator** of either {`start`,`end`} 4) an **explicit event** also in the form of a phrase, and 5) a **temporal relation** of either {`before`,`after`} that marks the relationship in the dimension defined by the *comparator* between the *implicit-event* and the *explicit-event*. With these 4 components, we are able to generate TE-style instances, using the context story as the premise, and "*implicit-event*, *comparator*, *temporal-relation*, *explicit-event*" as the positive (entailment) hypothesis. For example, in the first positive instance shown in Fig. 1, "distracted" is the *implicit-event*, "starts" is the *comparator*, "try" is *explicit-event* and "before" is the *temporal-relation*, leading to the hypothesis "distracted starts before try." We flip the *temporal-relation* (i.e., "before" to

[3]Refer to `http://leaderboard.allenai.org/tracie` for a public leaderboard and additional details.

Context Story (Premise)	**Hypothesis**	**Inference Label**
Tom needed to get braces. He was afraid of them. The dentist assured him everything would be fine. Tom had them on for a while. Once removed he felt it was worth it.	Tom avoids foods he can't eat with braces starts before the braces are removed.	`entailment`
We were all watching Spongebob as a family. It is a kid's show but all really enjoyed it. This one episode was especially funny for the adults. It has humor in it that is funny for kids and adults. It is something we can all watch...	The adults laughed at the jokes ends before we watch Spongebob as a family	`contradiction`
I was throwing the baseball with my son. He threw one past me that landed in the lake. I reached in to get the ball. I lost my balance and fell in. I got the ball and a bath all in one shot!	The ball was in the boys hand starts after he reached for the ball	`contradiction`

Figure 2: Example TRACIE instances. The **comparator** (i.e., {*starts,ends*}) and **relation** (i.e., {*before,after*}) in the hypothesis are highlighted, in addition to the corresponding explicit event from the story.

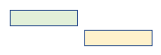
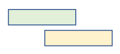
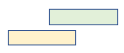
Illustration	Allen's Relation	Tracie's Relation
	Precedes, Meets	Starts Before Ends Before
	Overlaps, Finished-by, Contains, Starts, Equals, Started-by	Starts Before Ends After
	During, Finishes, Overlapped-by, Met-by, Preceded-by	Starts After Ends After

Figure 3: TRACIE's label definition and its relation to Allen's interval algebra, with a graph illustration between an implicit event and an explicit event.

"after" and vice versa) to create negative (contradiction) instances, as shown in the second example instance in Fig. 1.

Since the start times of *explicit-events* are more obvious to human annotators, we use them as reference points and compare the *implicit-event*'s start or end time with them (depending on the *comparator*), as the label definitions shown in Fig. 3. In rare cases where two time points are the same (e.g., *hit* and *get hit* start at the same time in Fig.1), we use the causal relation to decide the order, so that *hit* starts before *get hit*.

Such instances are created through a multi-stage annotation process where annotators are first asked to *write implicit events* related to explicit events in narrative stories. We then *automatically generate* the instances (without labels) using the implicit events, which is followed by a *label collection* step that annotates *temporal-relations*. All steps are implemented using the CrowdAQ platform (Ning et al., 2020a). We present the steps used to build these instances next.

Implicit Event Generation We randomly sample short commonsense stories from the ROCStories dataset (Mostafazadeh et al., 2016) and present each story to a human annotator. The annotator is asked to write 5 implicit event phrases that are not explicitly mentioned by the given story, but are inferable and relevant. For each of the written implicit events, we ask the annotators to write two explicit events closest to the implicit event's start and end time, respectively. With these two events, we can build two TRACIE instances (minus the *temporal-relation*) per implicit event, which accounts for 10 instances in total.

Automatic Instance Generation We use AllenNLP (Gardner et al., 2018) to extract all verbs and relevant arguments with its semantic role labeling (SRL) model. With all the verbs and their arguments, we can construct a pool of explicit events in the form of short phrases. For each of the implicit events, we randomly select two {*explicit-event*, *comparator*} pairs from the pool and build 10 additional instances (without *temporal-relation*).

Label Collection For each of the 20 instances per story, we annotate the *temporal-relation* with four different annotators. In this verification step, we follow the label definition in Fig. 3 and ask each annotator to produce a *temporal-relation* for each instance and use the majority agreement as the final label. We filter out the instances that are not able to produce a majority label.

2.2 Data Splits

To better understand how task-specific supervision contribute to models' performance, we propose two data splits, one using training/testing data from the same distribution, and another using differently sampled training and testing data.

IID Split We split the data under the independent and identically distributed (i.i.d.) assumption based on stories, with a 20/80 train/test ratio. The motivation for using a small training set, which follows Zhou et al. (2019), is that we believe temporal relations involve much commonsense knowledge.

As we later show in §5.3, it is infeasible to collect a large enough human-annotated training set to capture all the knowledge needed to tackle this problem completely, and a system must acquire knowledge from external resources. As a result, we use a small training set just to define the task, and at the same time, use an extensive testing set for more robust evaluation.

Easy/Hard Split In addition to the i.i.d. split, we propose another split called easy/hard. Recall that the annotators were asked to generate the closest explicit event to describe the implicit event's start or end time, as described in §2.1. These explicit events are often more obviously related to the implicit event. We consider such pairs to be "easy" for pre-trained language models (e.g., "I took the bus to school" and "I arrived at school" are closely related in their representations.[4]) In this split, we sample training instances only from these "easy" instances and create a 20/80 training/testing split. The goal is to test a system's ability to learn the task definitions instead of other potential artifacts, as the training distribution is much different from the testing distribution.

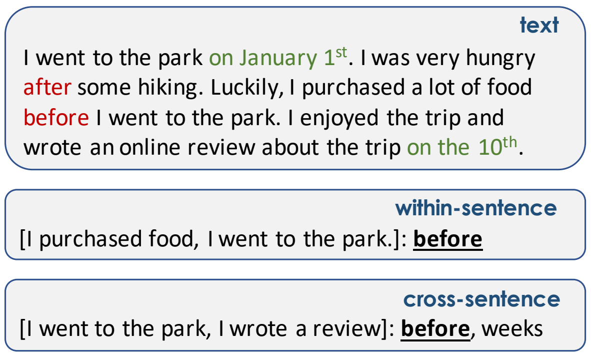
3 Pattern-Based Pre-Training

As argued in §2.2, it is nearly impossible to solve the temporal relation task by annotating a large number of training examples. We believe that it is more efficient to acquire temporal reasoning knowledge via distant supervision signals. The model learns the prior knowledge needed for the task and only subsequently learns the task definition through the small training set. This section describes how we collect the distant signals related to events' start times from large-scale texts. We use this extracted data to pre-train a model called *PatternTime*. While PatternTime will be used for fine-tuning directly on TRACIE, it will also form the basis of the more general modular temporal reasoning model called *SymTime* that we describe in §4.

3.1 Starting Point Comparison

We describe how we collect three sources of distant supervision to support the pre-training of a model that can predict the temporal ordering of events and estimate the time difference between events.

[4]The story may have other "hard" events, such as "I took a midterm", which requires reasoning over a chain of events.


text

I went to the park on January 1st. I was very hungry after some hiking. Luckily, I purchased a lot of food before I went to the park. I enjoyed the trip and wrote an online review about the trip on the 10th.

within-sentence

[I purchased food, I went to the park.]: **before**

cross-sentence

[I went to the park, I wrote a review]: **before**, weeks

Figure 4: Extraction for starting point comparison applied to an example paragraph.

Within-Sentence Extraction We collect start time comparisons between pairs of events heuristically from free-text using "before/after" keywords (following many prior works in temporal modeling and extraction (Do et al., 2012).) We use AllenNLP's SRL model to process each input sentence and find verbs with a temporal argument that starts with either "before" or "after", and contains at least one verb. If there are multiple verbs in the temporal argument, we take the one with the largest number of tokens as arguments. We match the two extracted verbs with the relation indicated by the first word of either "before" or "after".

As the example in Fig. 4 shows, the extractor identifies that *purchase food* is before *go to park* as indicated by the "before" keyword mentioned in the text. We apply this extraction process on the entire Wikipedia dump from May 2020, which produces 4 million pairs of comparisons.

Cross-Sentence Extraction The data collected from within-sentence patterns do not reveal how far the two starting points are from each other. As a result, we employ a cross-sentence extraction to find direct temporal mentions of hours and dates to get the distance. To do this, we use the SRL model to find all the temporal arguments and check for any direct mentions of time using syntactic rules. We apply this extraction process on the Wikipedia dump under the assumption that if a paragraph has many temporal arguments, the narrative order is the same as the events' temporal order. This is mostly true because the nature of Wikipedia is to explain situations as clear as possible. We can fill an unmentioned element of a temporal structure with the nearest previous mention with this assumption. For example, in Fig. 4, the second temporal expression only mentions "the 10^{th}", but we can inherit the previous mention of "January" and make a direct comparison between the two expressions. We normalize the difference between the two temporal expressions to one of seven coarse temporal units {≤minutes, hours, days, weeks, months, years, ≥decades}. As a result, we get *go to park* is **weeks** before *write review* as shown in Fig. 4.

We collect 300k pairs of events from this cross-sentence extraction process from Wikipedia. Each pair consists of two sentences containing the events to be compared, the original paragraph that contains the two sentences, a label of either *before* or *after* indicating the relationship between them, and a *distance* written as one of the 7 units above.

Original Masked LM Data We use the Gutenberg Dataset (Lahiri, 2014) to generate data for pre-training with the masked language model objective, to keep the original structure and most knowledge learned from the original pre-training. We collect 1 million paragraphs for this purpose.

Data Format For the first two sources, we format input sequences of `event: [EventA] starts [Relation] [EventB]. story: [Paragraph]` and output sequences of `answer: [Label] [Distance]`. Here `[EventA]` are the tokens that describe the first event; `[EventB]` are the ones that describe the second event; `[Paragraph]` are the tokens of the paragraph that contains both events. `[Relation]` is either `before` or `after` and `[Label]` is either `positive` or `negative`. When the label is positive, the relation will be the gold relation extracted from the text; when it is negative, the relation will be the opposite as the extracted relation. We randomly make 50% of the instances to be negative. `[Distance]` is one of the 7 coarse temporal units represented with a set of blank tokens `[extra_id_N]`. For the within-sentence extractions, we leave it to be blank so that the objective function will not include it when computing the loss.

3.2 Pattern-Based Distantly Supervised Model *(PatternTime)*

We use a sequence-to-sequence model as our base model, and pre-train it with the data collected in §3.1. We call the resulting model *PatternTime*, for pattern-based distantly supervised model for temporal relations.

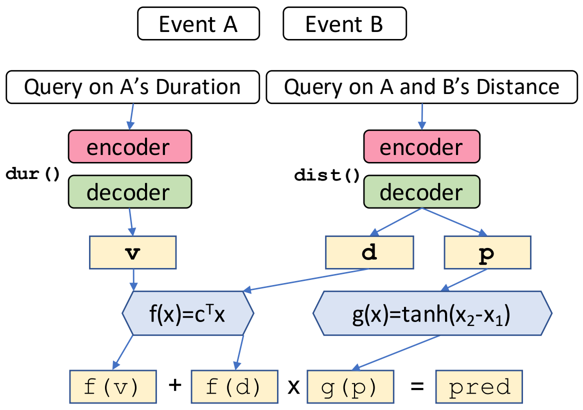


Figure 5: Overview of SymTime on end-time inferences.

4 Symbolic Reasoning Model (*SymTime*)

To address the challenge of predicting event end times for which it is difficult to obtain high-quality direct or distant supervision, we propose a reasoning model called *SymTime*. As shown in Fig. 5, this model infers end time by symbolically combining start time and duration from two separate predictions; such an approach takes inspiration from work on modular and decomposition-based modeling (Andreas et al., 2016; Khashabi et al., 2018; Khot et al., 2020). This section describes SymTime.

4.1 Formulation

As described in §2.1, hypotheses in TRACIE consist of pair-wise comparisons between two events e_1 and e_2 using a *comparator* from $\{\texttt{start}, \texttt{end}\}$ and a *query-relation* of either *before* or *after*. We assume that e_1 has a start time $\mathbf{start_1}$ expressed as a single-value UNIX timestamp,[5] and an end time $\mathbf{end_1}$. Similarly, e_2 has $\mathbf{start_2}$ and $\mathbf{end_2}$ that correspond to its start/end time, respectively. Under this assumption, hypotheses with a `start` comparator ask a model to estimate a relationship between $\mathbf{start_1}$ and $\mathbf{start_2}$ and compare which one is larger; a hypothesis with a `end` comparator asks the system to compare $\mathbf{end_1}$ and $\mathbf{start_2}$, as our label definitions show in Fig. 3.

We use PatternTime and its sequence-to-sequence objective directly to learn start-time hypotheses. For end-time hypotheses, we use the rule $\mathbf{end_j} = \mathbf{duration_j} + \mathbf{start_j}$ in order to infer end times with the help of duration estimations (later described in §4.2), and PatternTime's outputs. If we define $\mathrm{dist}(e_1, e_2) = \mathbf{start_1} - \mathbf{start_2}$ and

[5] https://www.unixtimestamp.com

$\mathrm{dur}(e_1) = \mathbf{duration_1}$, we can write the inference rule on hypotheses with `end` comparators as:

$$\begin{aligned}&\text{label} = \textit{before}\\ &\Leftrightarrow \mathbf{end_1} < \mathbf{start_2}\\ &\Leftrightarrow \mathbf{start_1} + \mathbf{duration_1} < \mathbf{start_2}\\ &\Leftrightarrow \big(\mathbf{start_1} - \mathbf{start_2}\big) + \mathbf{duration_1} < 0\\ &\Leftrightarrow \mathrm{dist}(e_1, e_2) + \mathrm{dur}(e_1) < 0\end{aligned}$$

Similarly, we have

$$\text{label} = \textit{after} \Leftrightarrow \ \mathrm{dist}(e_1, e_2) + \mathrm{dur}(e_1) > 0$$

One drawback of this inference rule is that it does not predict causal relations and therefore, cannot handle instances where $\mathbf{end_1} = \mathbf{start_2}$, because of our label definitions described in §2.1. We leave this problem for future research.

4.2 Duration Estimation

To obtain a model to estimate $\mathrm{dur}()$, we employ a pre-training process over the duration data from Zhou et al. (2020), which is similarly collected from pattern-based extraction. The data contains over 1 million events with their corresponding duration values, and we map each instance to an input sequence of `event: [Event]` and a corresponding output sequence `answer: [Value]`, where `[Event]` are the tokens of an event with the trigger verb marked by a special token to its left, and `[Value]` is one of the same 7 coarse unit labels as described in §3.1. We use the same base sequence-to-sequence model as we use in PatternTime to pre-train such a duration model.

4.3 Computation and Learning

We use the output from PatternTime to approximate the function $\mathrm{dist}()$. Following the sequence formulation of PatternTime in §3, we replace `[EventA]` with e_1, `[EventB]` with e_2, `[Paragraph]` with the context (premise) and fix `[Relation]` to be *before*. By taking the logits of "positive" and "negative" in `[Label]` and applying a softmax operation, we are able to get the probabilities $P_{\textit{before}}, P_{\textit{after}}$ for the starting points ordering. Similarly, we use the logits over the 7 vocabularies representing the units in `[Distance]` with softmax to get 7 probabilities $P_{\leq minutes}, P_{hours}, ..., P_{\geq decades}$ that estimate the absolute distance. We write the probabilities as two vectors $\mathbf{p} = [P_{\textit{before}}, P_{\textit{after}}], \mathbf{d} = [P_{\leq minutes}, ..., P_{\geq decades}]$.

To get a single value to represent $\mathbf{start_1} - \mathbf{start_2}$, we dot product the probabilities with an incremental constant vector $\mathbf{c} = [0, 1, 2, 3, 4, 5, 6]$ to get a median estimation of the absolute time difference. In order to get the direction, we apply the $\tanh$ function to the difference between the probabilities in $\mathbf{p}$.[6] As a result, we have:

$$\begin{aligned}\mathrm{dist}(\cdot) &= \mathbf{start_1} - \mathbf{start_2}\\ &= \mathbf{c}^T\mathbf{d} * \tanh(\mathrm{INT}_{max} * (p_2 - p_1))\end{aligned} \tag{1}$$

We use the pre-trained model in §4.2 to approximate the function $\mathrm{dur}()$. Because the model is pre-trained with markers to the left of trigger verbs, we run a part-of-speech tagger on input event and add a marker to the left of the first verb. By taking the logits of the 7 vocabularies representing the units in the prediction of `[Value]` in §4.2 with softmax, we get a single vector of probabilities $\mathbf{v} = [v_1, v_2, ...v_7]$, where $v_1 = P(\mathrm{dur} \leq \textit{minutes}), v_2 = P(\mathrm{dur} = \textit{hours}), ..., v_7 = P(\mathrm{dur} \geq \textit{decades})$, respectively. We use the same constant vector to get

$$\mathrm{dur}(\cdot) = \mathbf{dur_1} = \mathbf{c}^T\mathbf{v} \tag{2}$$

For hypotheses with comparator `start`, we use PatternTime and its sequence-to-sequence objective to learn (i.e., by using `[Label]` directly as the prediction). For hypotheses where the comparator is `end`, we use the inference process in §4.1 and computation process in §4.3 to construct $l = [\mathbf{pred}, -\mathbf{pred}], \mathbf{pred} = \mathrm{dist}(e_1, e_2) + \mathrm{dur}(e_1)$. We find the *gold-temporal-relation* in each training instance and compute a two-class cross-entropy loss with l. The PatternTime that predicts `start` hypotheses shares weights with the one used in computing the logits l. The final model SymTime can also be used to predict TRACIE instances without any task-specific supervision benefited from the distant supervision.

5 Experiments

In this section, we detail our experimental setup in §5.1 and §5.2. We then discuss our main results in §5.3 and Tables 1-2 for the different splits first introduced in §2.2. To demonstrate the effectiveness of our new models, we also describe results of an experiment on a non-TRACIE temporal benchmark in §5.4 and Table 3.

[6] To ensure that tanh returns a value close to 1 or -1, we multiply the distance by a big number denoted as INT_{max}.

5.1 Baselines and Systems

We use T5-Large implemented in Wolf et al. (2019) as our base sequence-to-sequence model for both PatternTime and the duration model in §4.2 as it provides faster iterations compared to the larger variations. We use 32 batch size, 5e-3 learning rate and early stopping. We observe that PatternTime converges after 35k steps (1M instances) and the duration model converges after 80k steps (2.5M instances). We use 32 batch size, 5e-3 learning rate and run 50 epochs for all TRACIE experiments. We repeat each experiment three times with different random seeds and report the averaged result.

We use two sets of baselines: one involving only the same base model T5-Large, and another set using other architectures. For the first set, we compare with a single T5-Large that is fine-tuned on TRACIE's training set (**BaseLM**) to see the effect of our pattern-based pre-training strategy in §3. We also train a T5-Large model that is first fine-tuned on 20k start-time comparison instances from MATRES (**BaseLM-MATRES**) (Ning et al., 2018b) before fine-tuned on TRACIE, in order to see the effect of additional human supervision. For the second set, we compare with a **biLSTM** model (as used in Bowman et al. (2015)), **Roberta-Large** (Liu et al., 2019) and **T5-3B**, a larger variant of T5. All baseline models follow a standard TE setup.

These baselines are compared against our two main models: **PatternTime**, which pre-trained on the start-time distant supervision data described in §3 before being fine-tuned on TRACIE and the **SymTime** described in §4. We also report the results of SymTime used without any fine-tuning on TRACIE (**SymTime-ZeroShot**) to the see the effect of using only knowledge acquired during SymTime's modular pre-training steps.

5.2 Metrics

We measure system performances on TRACIE separately for the start-time hypotheses and end-time hypotheses. We also employ a story-wide exact match metric, which is the percentage of stories with all its related hypotheses answered correctly.

5.3 Results

Table 1 shows systems performances on TRACIE's i.i.d. setting. We observe that *PatternTime* improves on all metrics over the base language model, with over 7% on the starting point and 6% on story-wide exact match. It also outperforms *BaseLM-MATRES*, demonstrating how large-scale high-precision distant supervision benefits this task.

System	Start	End	All	Story
BiLSTM	55.0	60.5	58.0	9.6
Roberta-Large	78.9	72.5	75.4	23.0
T5-3B	78.9	72.2	75.2	21.5
BaseLM (T5-large)	73.4	69.3	71.2	18.6
BaseLM-MATRES	77.0	71.0	73.7	22.3
PatternTime (ours)	80.9	72.0	76.0	25.1
SymTime (ours)	**82.7**	**73.2**	**77.5**	**28.6**
SymTime-ZeroShot	76.7	68.5	72.2	21.1

Table 1: Performances on i.i.d. split, best numbers in **bold**. Zeroshot uses no TRACIE instance as supervision; BaseLM is T5-large; Story is the percentage of story-wide exact match.

With a duration model involved to infer the end time, *SymTime* further improves on all metrics, with 9%, 4%, and 10% gains over the base language model on start time, end time and story-wide exact match, respectively. *SymTime* can further improve the performance on start time hypothses over *PatternTime* even though they use the same model to predict start-time queries. This is because *PatternTime* was not designed to understand end time during the pre-training process, and fine-tuning it on such data hurts the representation in general. *SymTime-ZeroShot*, which uses no task-specific supervision, can achieve on-par if not better overall performance comparing to the supervised BaseLM.

System	Start	End	All
BaseLM	53.5	69.3	62.4
PatternTime	69.0	**73.8**	71.7
SymTime	69.4	73.6	71.8
SymTime-ZeroShot	**77.3**	72.7	**74.7**

Table 2: Performances on easy/hard split (without story-wide exact match since this set is not split by story).

Table 2 compares the performances on the easy/hard split of TRACIE. Different from what we see in Table 1, all models fine-tuned on the "easy" training data suffer from training biases and artifacts, suggesting that none of the models correctly learns the task definition from the fine-tuning. This results in lower test performance, with *SymTime* achieving only 71.8% comparing to the 77.5% in i.i.d. setting. Compared to BaseLM, our proposed models cope with this training/testing distribution difference better because of a better

System	OT-NS	OT	OT-MS	PT
Wang et al. (2020)	85.9	-	-	-
BaseLM	86.0	87.5	77.4	69.0
SymTime	87.5	89.7	85.6	75.6

Table 3: Performances on MATRES. The result from Wang et al. (2020) is not strictly comparable with the rest.

temporal representation. This is also suggested by *SymTime-ZeroShot*, which achieves the best averaged-performance without any task-specific fine-tuning, as it is not affected by any potential training biases.

5.4 Extrinsic Evaluation

To show that our model is not specifically designed for the TRACIE dataset and is general in temporal relation reasoning, we also evaluate on MATRES (Ning et al., 2018b), a temporal relation dataset focused on comparing explicit events' starting points. We train and evaluate only the instances with a label of either "before" or "after", which accounts for 80% of all instances in MATRES.

We show the performance of SymTime, which is virtually the same as PatternTime, as MATRES does not evaluate duration or end time. In addition to the base language model, we compare with a SOTA system on MATRES from Wang et al. (2020). This system is trained on the original four labels, but we take the argmax between "before" and "after" probabilities and limit the output space to 2 labels. This makes the number not strictly comparable with our results, but we can get a sense of other systems' performances on this task.

We report four results - "OT-NS" stands for "original test, no story", which is when a model is trained and tested with only the sentences containing the trigger verbs. "OT" is trained and tested with the entire document as an auxiliary input. Here the story is downsampled by randomly removing sentences until the number of tokens drops below 512, the maximum sequence length of our models. "OT-MS (original test, minimal supervision)" is trained with 1.2k training instances (6% of the full training data), a similar amount comparing to TRACIE's i.i.d. split. "PT" stands for "perturbed test", where we train the models on the complete training set and test on a perturbed test set from Gardner et al. (2020).

Table 3 shows the performances of our proposed model and the baselines. We see that our model is consistently better than BaseLM, and at the same time, comparable to if not improves over a SOTA model (Wang et al., 2020). Our model only drops 4% when trained with less than one-tenth of the data, comparing to the 10% drop for T5-Large. This shows the benefit and effectiveness of task-related distant supervision.

6 Related Work

Temporal reasoning has received much attention in the NLP community. Initiated by TimeBank (Pustejovsky et al., 2003), people have proposed many datasets that touch on different aspects of temporal relations (Bethard et al., 2007; Cassidy et al., 2014; Reimers et al., 2016; O'Gorman et al., 2016; Ning et al., 2018b, 2020b), as well as benchmarks that focus on other temporal knowledge (Pan et al., 2006; Zhou et al., 2019). Multiple systems have been proposed in the temporal domain in order to solve temporal relation extraction (Do et al., 2012; Moens and Leeuwenberg, 2017; Leeuwenberg and Moens, 2018; Meng and Rumshisky, 2018; Ning et al., 2018c; Han et al., 2019), duration prediction (Vashishtha et al., 2019) and other tasks. Our decision to use a textual entailment style follows much of the recent work on natural language inference (Williams et al., 2017; Nie et al., 2020; Bhagavatula et al., 2020), which tends to not focus on temporal reasoning proper.

Commonsense tasks often require knowledge from distant supervision. Gusev et al. (2011) use syntactic patterns to acquire duration information, Ning et al. (2018a) acquires noisy global signals and Zhou et al. (2020) applies a large-scale extraction for language model pre-training. Our approach is unique, in that we do extraction over larger windows of text to capture longer-range dependencies in narrative text (for related ideas, see Ammanabrolu et al. (2020)).

Temporal knowledge is sparse in text and hard to be learned directly. As a result, people have tried to use structured approaches that use temporal constraints or connections between relations (Ning et al., 2017; Do et al., 2012). Neural module networks (Andreas et al., 2016; Gupta et al., 2019) define symbolic operations in a reasoning process, which is related to the intuition we apply in this work. Our work also takes inspiration from other neural modular, or decomposition-based approaches (Talmor and Berant, 2018; Wolfson et al., 2020; Khot et al., 2020) and the consistency logic framework in Li et al. (2019) in using differentiable symbolic computations in some of our neural models. In the latter case, we build neural-symbolic models that operationalize some of the classical interval-based computations used in earlier work on temporal reasoning (Allen, 1983; Gerevini and Schubert, 1995).

7 Conclusion

We propose a challenging temporal relation dataset TRACIE, to evaluate systems' understanding of implicit events and their start and end times. We propose a distant supervision process that improves language models' generalization of temporal relations. We further combine such a distantly supervised model that compares events' start times with another one that estimates events' duration. The final model uses the information to infer events' end times, which is hard to be directly extracted and predicted. We show that our model improves over multiple metrics on TRACIE and MATRES, suggesting the effectiveness of high-precision pretraining and symbolic temporal reasoning. The room for improvements also motivates future work on general temporal understanding.

References

James F Allen. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

Prithviraj Ammanabrolu, Wesley Cheung, William Broniec, and Mark O Riedl. 2020. Automated Storytelling via Causal, Commonsense Plot Ordering. *arXiv preprint arXiv:2009.00829*.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and D. Klein. 2016. Neural module networks. *CVPR*.

Steven Bethard, James H. Martin, and Sara Klingenstein. 2007. Timelines from Text: Identification of Syntactic Temporal Relations. *ICSC*.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Wen-tau Yih, and Yejin Choi. 2020. Abductive Commonsense Reasoning. *In ICLR*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A Large Annotated Corpus for Learning Natural Language Inference. *EMNLP*.

Taylor Cassidy, Bill McDowell, Nathanel Chambers, and Steven Bethard. 2014. An Annotation Framework for Dense Event Ordering. In *ACL*.

Quang Do, Wei Lu, and D. Roth. 2012. Joint Inference for Event Timeline Construction. In *EMNLP-CoNLL*.

Matt Gardner, Yoav Artzi, Victoria Basmova, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hanna Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, R. Tsarfaty, Eric Wallace, A. Zhang, and Ben Zhou. 2020. Evaluating Models' Local Decision Boundaries via Contrast Sets. *Findings of EMNLP*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew E. Peters, M. Schmitz, and L. Zettlemoyer. 2018. AllenNLP: A Deep Semantic Natural Language Processing Platform. *NLP-OSS*, abs/1803.07640.

Alfonso Gerevini and Lenhart Schubert. 1995. Efficient algorithms for qualitative reasoning about time. *Artificial intelligence*, 74(2):207–248.

Nitish Gupta, Kevin Lin, Dan Roth, Sameer Singh, and Matt Gardner. 2019. Neural Module Networks for Reasoning over Text. *In ICLR*.

Andrey Gusev, Nathanael Chambers, Divye Raj Khilnani, Pranav Khaitan, Steven Bethard, and Dan Jurafsky. 2011. Using Query Patterns to Learn the Duration of Events. In *IWCS*.

Rujun Han, Qiang Ning, and Nanyun Peng. 2019. Joint Event and Temporal Relation Extraction with Shared Representations and Structured Prediction. In *EMNLP*.

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2018. Question Answering as Global Reasoning over Semantic Abstractions. In *AAAI*.

Tushar Khot, Daniel Khashabi, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2020. Text Modular Networks: Learning to Decompose Tasks in the Language of Existing Models. *arXiv preprint arXiv:2009.00751*.

Shibamouli Lahiri. 2014. Complexity of Word Collocation Networks: A Preliminary Structural Analysis. In *Proceedings of ACL-SRW*.

Alice Lai, Yonatan Bisk, and Julia Hockenmaier. 2017. Natural Language Inference from Multiple Premises. *IJCNLP*.

A. Leeuwenberg and Marie-Francine Moens. 2018. Temporal information extraction by predicting relative time-lines. *EMNLP*.

Tao Li, Vivek Gupta, Maitrey Mehta, and V. Srikumar. 2019. A logic-driven framework for consistency of neural models. In *EMNLP*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv*, abs/1907.11692.

Yuanliang Meng and Anna Rumshisky. 2018. Context-aware neural model for temporal information extraction. In *ACL*.

Marie-Francine Moens and A. Leeuwenberg. 2017. Structured learning for temporal relation extraction from clinical records. In *EACL*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A Corpus and Evaluation Framework for Deeper Understanding of Commonsense Stories. *NAACL*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A New Benchmark for Natural Language Understanding. *In ACL*.

Qiang Ning, Z. Feng, and D. Roth. 2017. A Structured Learning Approach to Temporal Relation Extraction. In *EMNLP*.

Qiang Ning, H. Wu, H. Peng, and D. Roth. 2018a. Improving Temporal Relation Extraction with a Globally Acquired Statistical Resource. In *NAACL*.

Qiang Ning, Hao Wu, Pradeep Dasigi, Dheeru Dua, Matt Gardner, Robert L. Logan IV, and Zhen Nie. 2020a. Easy, Reproducible and Quality-Controlled Data Collection with CROWDAQ. *EMNLP*.

Qiang Ning, Hao Wu, Rujun Han, Nanyun Peng, Matt Gardner, and Dan Roth. 2020b. Torque: A reading comprehension dataset of temporal ordering questions. *EMNLP*.

Qiang Ning, Hao Wu, and Dan Roth. 2018b. A multi-axis annotation scheme for event temporal relations. In *ACL*.

Qiang Ning, Ben Zhou, Z. Feng, H. Peng, and D. Roth. 2018c. CogCompTime: A Tool for Understanding Time in Natural Language. In *EMNLP*.

Tim O'Gorman, Kristin Wright-Bettner, and Martha Palmer. 2016. Richer event description: Integrating event coreference with temporal, causal and bridging annotation. In *CNS*.

Feng Pan, Rutu Mulkar-Mehta, and Jerry R Hobbs. 2006. Extending TimeML with Typical Durations of Events. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*.

James Pustejovsky, José M Castano, Robert Ingria, Roser Sauri, Robert J Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R Radev. 2003. TimeM: Robust Specification of Event and Temporal Expressions in Text. In *New Directions in Question Answering*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-text Transformer. *JMLR*, 21(140):1–67.

Nils Reimers, Nazanin Dehghani, and Iryna Gurevych. 2016. Temporal Anchoring of Events for the Timebank corpus. In *ACL*.

Alon Talmor and Jonathan Berant. 2018. The Web as a Knowledge-base for Answering Complex Questions. *In NAACL*.

Siddharth Vashishtha, Benjamin Van Durme, and Aaron Steven White. 2019. Fine-grained Temporal Relation Extraction. *ACL*.

Haoyu Wang, Muhao Chen, Hongming Zhang, and Dan Roth. 2020. Joint constrained learning for event-event relation extraction. *EMNLP*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *In NAACL*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it Down: A Question Understanding Benchmark. *TACL*, 8:183–198.

Ben Zhou, Daniel Khashabi, Qiang Ning, and D. Roth. 2019. "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. In *EMNLP*.

Ben Zhou, Qiang Ning, Daniel Khashabi, and Dan Roth. 2020. Temporal Common Sense Acquisition with Minimal Supervision. In *ACL*.