Temporal Validity Change Prediction

Anonymous ACL submission

Abstract

Temporal validity is an important property of text that is useful for many downstream applications, such as recommender systems, conversational AI, or story understanding. Existing benchmarking tasks often require models to identify the temporal validity duration of a single statement. However, in many cases, additional contextual information, such as sentences in a story or posts on a social media profile, can be collected from the available text stream. This contextual information may greatly alter the duration for which a statement is expected to be valid. We propose *Temporal Validity Change Prediction*, a natural language processing task benchmarking the capability of machine learning models to detect contextual statements that induce such change. We create a dataset consisting of temporal target statements sourced from Twitter and crowdsource sample context statements. We then benchmark a set of transformer-based language models on our dataset. Finally, we experiment with temporal validity duration prediction as an auxiliary task to improve the performance of the state-of-the-art model.

1 Introduction

In human communication, temporal properties are frequently underspecified when authors assume that the recipient can infer them via commonsense reasoning. For example, when reading "I am moving on Saturday", a reader is likely to assume the person will be busy for most of the day. On the other hand, when reading "I will make a sandwich on Sunday", this is likely to only take up a fraction of the author's day and may not impact other plans. Such reasoning is referred to as *temporal commonsense* (TCS) *reasoning* (Wenzel and Jatowt, 2023).

Temporal validity (Almquist and Jatowt, 2019; Hosokawa et al., 2023; Lynden et al., 2023) is a property that is vital for the proper understanding of a text. The temporal validity of a statement, i.e., whether it contains valid information at a given time, often requires TCS reasoning to resolve. For example, in determining whether a statement like "I am driving home from work" is still valid after five hours, we may use our prior understanding of the typical duration of related events, such as commuter traffic. While the amount of research into TCS and, to a degree, temporal validity, has risen over the past years (Wenzel and Jatowt, 2023), there are still several properties of temporal validity that have not been considered in previous research. One such property is the impact of *context* on the temporal validity duration of a statement. For example, the sentence "I am driving home from work" may be valid for a longer time when followed by a statement such as "There is a massive traffic jam".

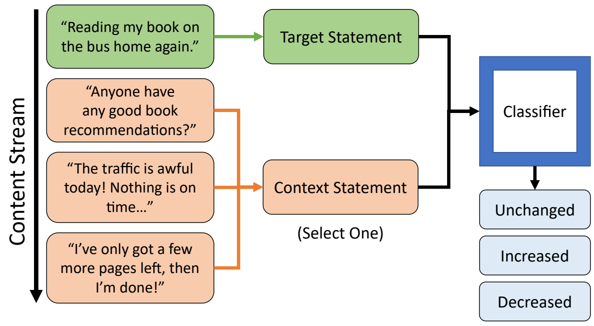

Figure 1: A visualization of the TVCP task

To model this problem, we propose a new NLP task format called *Temporal Validity Change Prediction* (TVCP), which requires reasoning over whether a context statement changes the temporal validity duration of a target statement. The task is visualized in Figure 1. We propose the following applications for such a system.

Timeline Prioritization: Social media services such as Twitter rely on recommender systems to prioritize the vast amount of content that their users produce. One possible way to improve the prioritization of content is to consider its temporal validity (Takemura and Tajima, 2012; Koul et al., 2022), as users are likely to care about current and relevant information over more general, stationary statements. TVCP can be used to leverage the stream of social media posts by a given user as possible context to better estimate the temporal validity duration of a previously observed post.

User Status Tracking: Similarly, the content of a user's posts on social media could be utilized for other analytical or business purposes, such as predicting revenue streams (Asur and Huberman, 2010; Deng et al., 2011; Lassen et al., 2014; Lu et al., 2014) or identifying trends in a community's or an individual user's behaviour (Li et al., 2018; Abe et al., 2018; Shen et al., 2020). TVCP could be used to identify posts that refer to previous temporal information, to detect chains of thought about topics that may not be self-contained.

Conversational AI: Foundation models, such as CHATGPT (Ouyang et al., 2022) and BARD (Manyika, 2023), could use the temporal validity of statements provided by the user to keep track of knowledge that is still relevant to the conversation. Using TVCP, new messages by the user could be evaluated to adjust the expected temporal validity period of previously learned facts. This is especially relevant as initial reports indicate that foundation models may struggle with TCS reasoning (Bian et al., 2023).

Our main contributions are the following:

1. We define a novel NLP task (TVCP). This ternary classification task requires models to predict the impact of a context statement on a target statement's temporal validity duration.

2. We build a dataset of tuples consisting of time-sensitive *target statements*, as well as *follow-up statements* that act as context for our task.

3. We evaluate the performance of existing pre-trained *language models* (LMs) on our dataset, including models fine-tuned on other TCS tasks as well as CHATGPT.

4. We propose an augmentation to the training process that leverages temporal validity duration information to help improve the performance of the state-of-the-art classifier.

2 Related Work

2.1 Temporal Commonsense Reasoning

TCS reasoning is often considered one of several categories of commonsense reasoning (Storks et al., 2019a; Bhargava and Ng, 2022). A major driver of research specifically into TCS appears to have been the transformer architecture (Vaswani et al., 2017) and resulting LMs. In recent years, several datasets that specifically aim to benchmark TCS understanding have been published (Zhou et al., 2019; Ning et al., 2020; Zhang et al., 2020; Qin et al., 2021; Zhou et al., 2021), while ROCSTORIES (Mostafazadeh et al., 2016) appears to be the only dataset focussing on this type of reasoning before the publication of the transformer architecture. Small adjustments to transformer-based LMs are often proposed as state-of-the-art solutions for these datasets (Pereira et al., 2020; Yang et al., 2020; Zhou et al., 2020; Pereira et al., 2021; Kimura et al., 2021; Zhou et al., 2021, 2022; Cai et al., 2022; Yu et al., 2022). Similarly, temporalized transformer models are popular solutions for tasks such as document dating or semantic change detection (Rosin and Radinsky, 2022; Rosin et al., 2022; Wang et al., 2023).

The TCS taxonomy defined by Zhou et al. (2019) is frequently referenced. It contains the five dimensions of *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).

2.2 Temporal Validity

Compared to TCS reasoning, temporal validity in text is a less well-researched field. It effectively combines three dimensions of the taxonomy by Zhou et al. (2019): *Stationarity*, to reason about whether a statement contains temporal information, *typical time*, to reason about when the temporal information occurs, and *duration*, to reason about how long the temporal information takes to resolve.

Takemura and Tajima (2012) classify the lifetime duration of tweets, i.e., the informational value of a tweet over time. They use handcrafted, domain-specific features to train a *support vector classifier* (SVC) on supervised samples.

Almquist and Jatowt (2019) similarly design features to estimate the temporal validity duration of sentences collected from news, blog posts, and Wikipedia using SVCs. Their features contain general properties such as the word- or sentence length, but also more complex ones, such as latent semantic analysis.

Method	Task	Data Source	Duration Bias	Model	# Samples
Takemura and Tajima (2012)	TV_d	Twitter	N/A	SVC	9,890
Almquist and Jatowt (2019)	TV_d	Blogs, News, Wikipedia	*years*	SVC	1,762
Hosokawa et al. (2023)	TNLI	Image Captions	*seconds*[1]	LM	10,659
Lynden et al. (2023)	TV_d	WikiHow	*hours*	LM	339,184
Ours	TVCP	Twitter	*hours*	LM	5,055

Table 1: Summary of related work

Hosokawa et al. (2023) define the *Temporal Natural Language Inference* (TNLI) task. The goal of TNLI is to determine whether the temporal validity of a given hypothesis sentence is supported by a premise sentence.

Lynden et al. (2023) build a large dataset of human annotations specifying the duration required to perform various actions on WikiHow as well as their respective temporal validity durations.

2.3 Comparison with Related Work

Table 1 shows the most closely related research. As noted, our dataset is based on the proposed TVCP task, whereas previous work was based on the TV_d and TNLI tasks. All tasks are described in more detail in Section 3.

Another prominent distinctive attribute is the text source and the resulting temporal validity duration bias. For example, sentences sourced from news or Wikipedia articles often appear to be valid for years or longer. On the other hand, image captions may only be valid for a few seconds or minutes. We decided to source our sentences from Twitter due to its alignment with our downstream use cases. Similar to Lynden et al. (2023), our collected temporal information tends to be valid for a few hours.

We follow recent research by evaluating our dataset using transformer-based LMs, whereas earlier approaches relied on methods such as SVCs.

Except for the CoTAK dataset (Lynden et al., 2023), the datasets tend to be relatively small. As crowdsourcing is used in all datasets referenced in Table 1 to annotate text spans with commonsense information, the costs of dataset creation can quickly escalate. In addition, we ask participants to create examples of follow-up statements that cause temporal validity change. This approach further restricts the overall size of our dataset due to the relative difficulty of the task.

[1]Based on analysis of a sample. TV_d labels are not available for the full dataset.

3 Task

3.1 Defining Temporal Validity

Temporal validity, in essence, is simply the time-dependent validity of a text. As shown in Equation 1, the temporal validity of a statement s at a time t is a binary value that determines whether the information in s is valid at the given time.

$$\mathrm{TV}(s,t) = \begin{cases} \text{True} & \text{if information in } s \text{ is valid at } t, \\ \text{False} & \text{otherwise} \end{cases} \quad (1)$$

In some previous research (Hosokawa et al., 2023; Lynden et al., 2023), the scope of evaluated temporal information is limited to actions, such as "I am *baking bread*". However, we note that other types of temporal information exist, such as events (e.g., in the sentence "*Job interview* tomorrow") or temporary states (e.g., in the sentence "It *is nice out* today"). In an analysis of a subset of our collected statements, shown in Figure 2, we find that these alternative types of temporal information constitute a significant portion (28%) of samples. Additionally, one-third of sampled statements contained at least two distinct pieces of temporal information with differing temporal validity spans. This indicates that the true scope of determining the temporal validity of a text may exceed what current datasets are benchmarking.

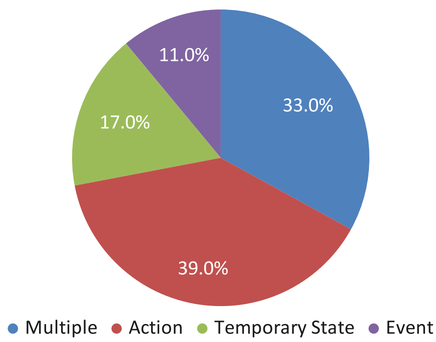

Figure 2: Distribution of different types of temporal information in a sample of our dataset

We assume that the temporal validity of stationary statements is constant for any given timestamp t. A stationary statement may be continuously true (e.g., "Japan lies in Asia"), or continuously false (e.g., "Japan lies in Europe"). This includes information that is fully contained in the past (e.g., "I went to the bank yesterday"). In general, we do not expect the validity of such a statement to change.

For contemporary or future information, we assume the statement is valid from the moment of sentence conception until the information is no longer ongoing. We include the duration of the information, rather than just its occurrence time, as humans are likely to still consider durative information relevant while it is ongoing. For example, we may reason that the statement "I will take a shower at 8 p.m." still has informational value at 8:05 p.m., as it allows us to infer the current action of the author.

3.2 Formalizing Existing Tasks

3.2.1 Temporal Validity Duration Estimation

Temporal Validity Duration Estimation (TV_d) is the primary task that is evaluated in temporal validity research (Takemura and Tajima, 2012; Almquist and Jatowt, 2019; Lynden et al., 2023). The goal is to estimate the duration for which the statement is valid, starting at the statement creation time. We formalize this task in Equation 2, where t_s is the timestamp at which the statement s is created.

$$\mathrm{TV}_d(s) = \max_{t \geq t_s}\{t \mid \mathrm{TV}(s,t) = \mathrm{True}\} \tag{2}$$

The TV_d task is useful in downstream applications such as social media, where information on the posting time of a statement is readily available and can be used to infer the span during which the statement is valid.

3.2.2 Temporal Natural Language Inference

The goal of TNLI (Hosokawa et al., 2023) is to infer whether a statement is temporally valid, given additional context, using typical NLI terminology (MacCartney, 2009; Storks et al., 2019b). TNLI requires a *hypothesis statement* (that we call *target statement*, or s_t) and a *premise sentence* (that we call *follow-up statement*, or s_f). Implicitly, the inference takes place at t_{s_f}, that is, the posting time of the follow-up statement, but no explicit duration information is required to solve this task. Formally, we define TNLI in Equation 3 (SUO = supported, INV = invalidated, UNK = unknown), where $\mathrm{TV}^c(s,t)$ is the temporal validity of a statement s at a time t given context c. The UNK class is assigned in cases where $\mathrm{TV}^{s_f}(s_t, t_{s_f})$ is neither clearly supported nor invalidated by the context.

$$\mathrm{TNLI}(s_t, s_f) = \begin{cases} \mathrm{SUO} & \mathrm{TV}^{s_f}(s_t, t_{s_f}) = \mathrm{True} \\ \mathrm{INV} & \mathrm{TV}^{s_f}(s_t, t_{s_f}) = \mathrm{False} \\ \mathrm{UNK} & \mathrm{TV}^{s_f}(s_t, t_{s_f}) = \mathrm{Unclear} \end{cases} \tag{3}$$

Unlike TV_d, this task format lends itself to downstream applications such as story understanding, wherein a larger text stream of individual statements is provided with no clear explicit notion of time passing between each sentence (e.g., in a book).

3.3 Temporal Validity Change Prediction

We propose *Temporal Validity Change Prediction* (TVCP), which combines ideas from both the inference- and duration-based tasks. Like TNLI, we require s_t and s_f for classification, and determine a ternary label that provides information about the impact of s_f on s_t. Unlike TNLI, our goal is to predict a *change* in the temporal validity *duration* of s_t.

We consider TVCP a necessary step in accurately determining a statement's temporal validity. Simply estimating the duration of the statement alone may not yield very precise results when it is, as in many use cases, extracted from a rich context, such as a book, a story, a news article, a step-by-step guide, or a social media profile. In these cases, surrounding information may provide additional context that could lead us to a different TV_d estimate. Simply concatenating s_t and s_f may lead to the classification of temporal information within s_f, which is undesired. Our segmentation of s_t and s_f into different semantic roles, similar to TNLI, prevents this issue.

Formally, we define TVCP in Equation 4 (DEC = decreased, UNC = unchanged, INC = increased), where $\mathrm{TV}_d^c(s)$ is the temporal validity duration of a statement s given context c. Figure 3 shows a concrete comparative example of the goal of all three tasks.

$$\mathrm{TVCP}(s_t, s_f) = \begin{cases} \mathrm{DEC} & \mathrm{TV}_d(s_t) > \mathrm{TV}_d^{s_f}(s_t) \\ \mathrm{UNC} & \mathrm{TV}_d(s_t) = \mathrm{TV}_d^{s_f}(s_t) \\ \mathrm{INC} & \mathrm{TV}_d(s_t) < \mathrm{TV}_d^{s_f}(s_t) \end{cases} \tag{4}$$

Since TVCP is a signal measuring the difference between TV_d with- and without s_f, respectively, a

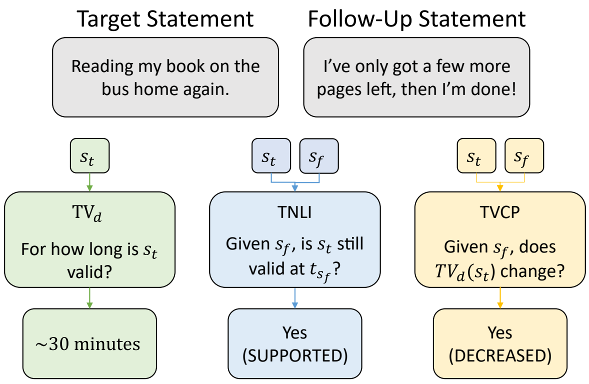

Figure 3: An example of TV_d, TNLI and TVCP

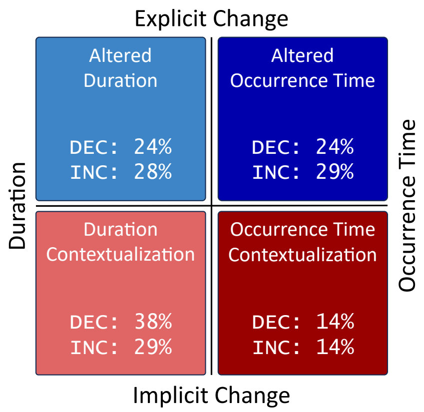

Figure 4: Dimensions of temporal validity change. The frequency of each category for DEC and INC classes in our sample is appended.

more fine-grained TV_d classification increases the number of TVCP instances that can be detected. On the other hand, evaluating TV_d on a very fine-grained scale may be more difficult for both models and humans (Honda et al., 2022), and the resulting uncertainty and inaccuracies could lead to a degradation of the system as a whole.

In our sample analysis, we find that temporal validity change generally occurs along two axes, shown in Figure 4. The first dimension is ***implicit*** versus ***explicit*** change. For example, an appointment may be postponed, which is an explicit change. On the other hand, the author may note in a follow-up statement that the appointment is in a sleep laboratory, which may cause us to re-evaluate for how long the original statement is valid, although the information itself has not changed.

The second dimension is a change to the ***occurrence time*** versus the ***duration*** of the information. For example, a flight may be delayed, in which case the occurrence time changes. Alternatively, the flight might have to be re-routed mid-air due to bad weather, in which case the duration changes.

In our sample, we find that all four categories are present to a reasonable degree in both the DEC and INC classes. Generally, changes to the duration tend to be slightly more frequent than changes to the occurrence time. This makes sense, as the occurrence time is a dimension that is only present when the information occurs in the future, whereas the duration of temporal information can change irrespective of the occurrence time.

4 Dataset

We create a dataset for training and benchmarking TVCP, where each sample is a quintuple $< s_t, s_f, \mathrm{TV}_d(s_t), \mathrm{TV}_d^{s_f}(s_t), \mathrm{TVCP}(s_t, s_f) >$.

s_t is collected by querying the Twitter API for tweets with no external context (e.g., no tweets that are retweets or replies, or tweets containing media). We apply several pre-processing steps to remove tweets whose content may not be self-contained. We aim to minimize spam and offensive content by applying publicly available LMs and word-list-based filters. To decrease the number of stationary statements, we employ an ensemble classifier based on the ALMQUIST2019 (Almquist and Jatowt, 2019) and COTAK datasets and select the most likely statements to contain temporal information. Finally, crowdworkers can tag any remaining stationary samples during the annotation process. A summary of our pre-processing pipeline is shown in Figure 5. Our code, including all preprocessing steps, is published under the Apache 2.0 licence.

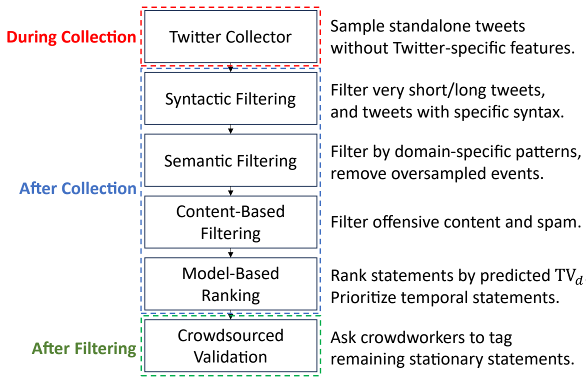

Figure 5: A summary of our tweet collection pipeline

For each target statement, we ask two crowdworkers to estimate $\mathrm{TV}_d(s_t)$ from the logarithmic class design shown in Equation 5, which is modelled after human timeline understanding (Jatowt and Au Yeung, 2011; Varshney and Sun, 2013; Howard, 2018). If the annotators disagreed, we supplied a third vote. We discarded any tweets that were annotated as *less than one minute*, *more than one month*, or *no time-sensitive information* (i.e., stationary), as well as tweets where no majority agreement could be reached. Of 2,996 annotated target tweets, 571 were discarded without a third annotation, 867 were added without a third annotation, 546 were discarded after providing a third vote, and 1,012 were added after providing a third vote. The distribution of resulting TV_d labels before temporal validity change is shown in Figure 6.

$$t \in \{< 1 \text{ minute}, 1\text{-}5 \text{ minutes}, 5\text{-}15 \text{ minutes}, 15\text{-}45 \text{ minutes}, 45 \text{ minutes-}2 \text{ hours}, 2\text{-}6 \text{ hours}, \text{more than } 6 \text{ hours}, 1\text{-}3 \text{ days}, 3\text{-}7 \text{ days}, 1\text{-}4 \text{ weeks}, \text{more than } 1 \text{ month}\} \tag{5}$$

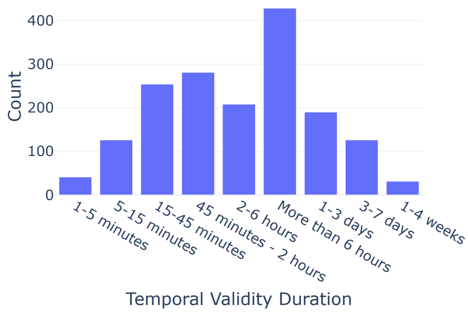

Figure 6: Distribution of TV_d labels (before temporal validity change) in our dataset

Both s_f and $\mathrm{TV}_d^{s_f}(s_t)$ were provided by a separate set of crowdworkers, given s_t and $\mathrm{TV}_d(s_t)$ as an input. In total, we collected 5,055 samples from 1,685 target statements. In Figure 7, we plot the *temporal validity change delta*, which is the class distance between the original and the updated TV_d estimate. We find that, in most cases, the temporal validity duration of a target statement is shifted only by one class.

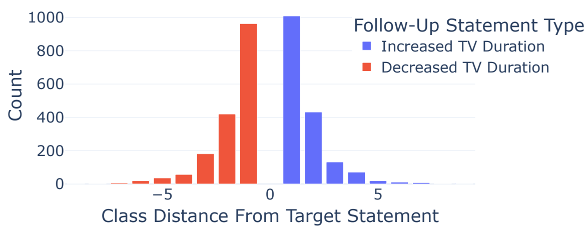

Figure 7: Temporal validity change delta distribution

All crowdsourcing tasks were set up on Amazon Mechanical Turk, using qualification tests, participation criteria, and manual verification of results to ensure high-quality samples (see Appendix A). We publish the resulting dataset for public use under the CC BY 4.0 licence. In accordance with the Twitter developer policy[2], we only publish the Tweet IDs of sourced statements. This also means original tweet authors retain the ability to delete their content, effectively removing it from the dataset.

5 Experiments

5.1 Language Models

We evaluate a set of transformer-based LMs on our dataset. We test four different archetypes in total:

- TRANSFORMERCLASSIFIER: Builds a hidden representation from the sentence-embedding token of the concatenation of s_t and s_f.
- SIAMESECLASSIFIER: Builds a hidden representation from the concatenated embeddings $[h_{s_t}, h_{s_f}, h_{s_t} - h_{s_f}, h_{s_t} \otimes h_{s_f}]$, where h_{s_t} and h_{s_f} are the sentence-embedding tokens of the target- and follow-up statement, respectively (Bromley et al., 1993; Nandy et al., 2020).
- SELFEXPLAIN (Sun et al., 2020): Builds a hidden representation from the embeddings of spans between arbitrary tokens in either s_t or s_f, selected by the model.
- CHATGPT: A chain-of-thought (Wei et al., 2022) reasoning prompt based on few-shot learning (one sample per TVCP class), passed to the gpt–3.5–turbo model via the OpenAI API. [3]

For the TRANSFORMERCLASSIFIER and SIAMESECLASSIFIER pipelines, we evaluate BERT-BASE-UNCASED (Kenton and Toutanova, 2019; 110M parameters) and ROBERTA-BASE (Liu et al., 2019; 125M parameters) embeddings. For SELFEXPLAIN, we only test the original implementation with ROBERTA-BASE embeddings. To evaluate transfer learning from other TCS tasks, we test the TRANSFORMERCLASSIFIER pipeline on regular BERT-BASE-UNCASED pre-training weights as well as two variants TACOLM (Zhou et al., 2020) and COTAK (Lynden et al., 2023),

[2] https://developer.twitter.com/en/developer-terms/policy, accessed 12.10.2023

[3] This call uses the most recent GPT3.5 model. We collected CHATGPT responses in July 2023.

which have the same underlying architecture, but use weights fine-tuned on existing TCS datasets.

We use the ADAMW optimizer (Loshchilov and Hutter, 2018) with $\varepsilon = $ 1e-8, $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight_decay $= 0.01$. We optimize for cross-entropy loss. SELFEXPLAIN adds an additional loss parameter in the form of squared span-attention weights, to encourage the model to more sharply choose which spans should be used to build the hidden representation.

We set the dropout probabilities and learning rates as defined in Table 2 as a result of our hyperparameter optimization (see Appendix C). For all models, the hidden embedding size is 768. For some ROBERTA-based models, we freeze embedding layers (i.e., only fine-tune intermediate and classification weights), as training all parameters leads to poor performance.

Model	Dropout	LR	Frozen
TF - BERT	0.25	1e-4	False
S - BERT	0.25	1e-4	False
TF - ROBERTA	0.25	1e-3	True
S - ROBERTA	0.10	1e-4	True
SELFEXPLAIN	0.00	2e-5	False

Table 2: Hyperparameter settings for different models. TF = TRANSFORMERCLASSIFIER, S = SIAMESECLASSIFIER

5.2 Multitask Implementation

For all archetypes except CHATGPT, we provide a second implementation, in which we add two regression layers that aim to respectively predict $\text{TV}_d(s_t)$ and $\text{TV}_d^{s_f}(s_t)$ from the same hidden representation. For these layers, we calculate the mean squared error between a single output neuron and a linear mapping of the TV_d class index to the range $[0, 1]$. Our intuition is that embeddings with an understanding of TV_d may be better suited for TVCP. Inspiration for this approach are models that utilize the interplay between temporal dimensions to improve the TCS reasoning performance in LMs, such as SYMTIME (Zhou et al., 2021) or SLEER (Cai et al., 2022).

5.3 Evaluation

We evaluate two metrics, accuracy and *exact match* (EM). Accuracy is simply the fraction of correctly classified samples. EM is the fraction of *target statements* for which all three samples were correctly classified. This metric punishes inconsistency in the model more strictly, thus providing a better view of the true performance and task understanding of each model (Wenzel and Jatowt, 2023), while disincentivizing shallow reasoning behaviours commonly seen in transformer models (Helwe et al., 2021; Tan et al., 2023).

We report the mean EM and accuracy across a five-fold cross-validation split. Each evaluation consists of 70% training data, 10% validation data, and 20% test data. If the validation EM does not exceed the best previously observed value for 5 consecutive epochs, we stop training. The model with the best validation EM is used for evaluation on the test set. The results are shown in Table 3.

Model	$\overline{\text{Acc}}$ (+ MT)	$\overline{\text{EM}}$ (+ MT)
TF - ROBERTA	64.0 (+1.5)	21.2 (+2.5)
CHATGPT	66.3 (N/A)	29.3 (N/A)
S - ROBERTA	78.7 (+1.1)	48.2 (+2.1)
TF - COTAK	83.2 (+0.6)	58.2 (+1.4)
S - BERT	83.8 (−0.3)	59.1 (−1.5)
TF - TACOLM	83.5 (+1.4)	59.1 (+2.9)
TF - BERT	84.8 (−0.2)	61.2 (+0.9)
SELFEXPLAIN	88.5 (+1.1)	69.8 (+2.8)

Table 3: Model evaluation results, sorted by mean EM score. TF = TRANSFORMERCLASSIFIER, S = SIAMESECLASSIFIER, MT = Multitask Implementation

We note a positive impact on the EM score from implementing multitasking in all models except the Siamese architecture with BERT-based embeddings. We use bootstrapping to calculate the statistical significance of implementing multitask learning on the best-performing model, SELFEXPLAIN, evaluating the number of bootstrap samples in which the multitask implementation outperforms regular SELFEXPLAIN. We find $p = 0.0027$ for accuracy, with a 95% confidence interval of $[0.0036, 0.0192]$. For EM, $p = 0.0025$, with a 95% confidence interval of $[0.0089, 0.0487]$.

The use of weights from other TCS tasks does not seem to have a positive impact on the performance of the TRANSFORMERCLASSIFIER pipeline. It is possible that, although the resulting embeddings are more aligned with temporal properties (Zhou et al., 2020), other important information in the embeddings is lost, leading to an overall decreased performance.

Due to some ROBERTA-based models having frozen embedding layers, the baseline performance by ROBERTA is much worse, but it improves much more when switching to the SIAMESECLASSIFIER implementation. We hypothesize that ROBERTA's sentence embedding token, <s>, may contain less information about the full sequence than BERT's [SEP] token due to the lack of a next-sentence-prediction task during pre-training.

CHATGPT ranks among the lower-performing models, which is consistent with other studies on TCS understanding (Bian et al., 2023). Its shortcomings may be due to the few-shot learning approach and a lack of knowledge about dataset specifics traits, which a trained classifier could leverage. Additionally, we do not specify our class design in the CHATGPT prompt, which could make it harder for CHATGPT to isolate the UNC class.

To evaluate the impact of training data quantity on classifier performance, we train our best-performing classifier (SELFEXPLAIN with multi-tasking, which we dub MULTITASK) on a single train-val-test split (80%/10%/10%) with different amounts of training data. The results can be seen in Figure 8. We find that performance increases as more data is used for training, but this effect starts to diminish as we approach 100% of our training data.

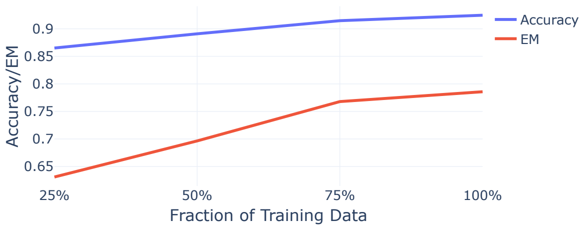

Figure 8: Training data vs. performance metrics in MULTITASK

In testing SELFEXPLAIN and MULTITASK on various temporal validity change deltas (Figure 9), we find they perform comparably on the UNC class, but MULTITASK slightly outperforms SELFEXPLAIN on all delta values greater than zero. While CHATGPT's subpar performance in the UNC class can partially be attributed to prompt design, it continues to lag far behind other models in the DEC and INC classes.

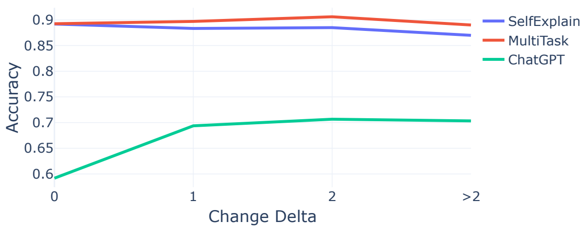

Figure 9: Temporal validity change delta vs. accuracy in MULTITASK, SELFEXPLAIN and CHATGPT

All models were trained and evaluated on an MSI GeForce RTX 3080 GAMING X TRIO 10G GPU using CUDA 11.7. Training and evaluation of all final models as well as hyperparameter tests took around 15 GPU hours.

6 Conclusion and Future Work

In this work, we have introduced TVCP, an NLP task, to aid in the accurate determination of the temporal validity duration of text by incorporating surrounding context. We create a benchmark dataset for our task and provide a set of baseline evaluation results for our dataset. We find that the performance of most classifiers can be improved by explicitly incorporating the temporal validity duration as a loss signal during training to improve the resulting embeddings. Despite the impressive feats performed by foundation models, we report, similar to previous work (Bian et al., 2023), poor performance in the TCS domain. These findings show that users should carefully evaluate whether a model like CHATGPT properly understands a given task before choosing it over smaller, fine-tuned LMs. We hypothesize that the performance of all models could further increase with additional training data.

Possible future work includes using the provided dataset and classifiers to collect a larger number of TVCP samples and annotating them with an updated temporal validity duration. A comparison of context-aware TV_d classifiers with prior models, like those by Almquist and Jatowt (2019), would shed light on the significance of accurate semantic segmentation between target and context. Similarly, the use of our dataset for generative approaches could be explored, for example, in the context of generative adversarial networks. For our multi-tasking implementation, directions for future work could be changes to hyperparameters such as the weight of the auxiliary loss, changes to the definition of the auxiliary task (e.g., log-scaled regression or ordinal classification), or even entirely new auxiliary tasks. Finally, current methods face limitations due to the effort of manual removal of stationary samples (Almquist and Jatowt, 2019; ours) or altering task definitions to avoid them (Hosokawa et al., 2023; Lynden et al., 2023). Research into models differentiating temporal and stationary information could enhance the development and definition of future TCS reasoning tasks.

Limitations

Although we focus on creating a reproducible training- and evaluation environment, some variables are out of our control. For example, bit-wise reproducibility is only guaranteed on the same CUDA toolkit version and when executed on a GPU with the same architecture and the same number of streaming multiprocessors. This means that an exact reproduction of the models discussed in this article may not be possible. Nevertheless, we expect trends to remain the same across GPU architectures.

The use of CHATGPT as an example of foundation model performance may be limiting due to its black box design. In the future, open-source models, such as LLAMA 2 (Touvron et al., 2023), could be evaluated to improve the reproducibility of foundation model performance claims. We chose to benchmark CHATGPT due to its common use as a baseline and in end-user scenarios, but the evaluation results may not be transferrable to other foundation models or even other versions of CHATGPT.

One of the major limitations of our approach is likely the dataset size. Although a relatively small dataset size is common in TCS reasoning, we find that our model performance still increases with the amount of training data used. The existing synthesized context statements in our dataset could be used to bootstrap an approach for automatically extracting additional samples from social media to alleviate this issue.

The data we collect is not personal in nature. However, the possibility of latent demographic biases in our data exists, for example, with respect to certain language structures or expressions used in the creation of follow-up statements. This could lead to the propagation of any such bias when the dataset is used to bootstrap further data collection, which should be considered in future work.

Our external validity is mainly threatened by two factors. First, our context statements are crowd-sourced. While we apply several steps to ensure the produced context is sensible, it is unclear whether downstream context, such as on social media platforms, manifests in similar structures as in our dataset, with respect to traits such as sentence length, grammaticality, and phrasing.

Second, similar to how pre-training weights from other TCS tasks do not seem to improve the classifier performance on our dataset, the weights generated as part of our training process are likely very task-specific, and may not generalize well to other tasks or text sources.

Overall, we recommend the use of the TVCP dataset and classifiers for bootstrapping further research into combining the duration- and inference-based temporal validity tasks, as well as research into directly predicting updated temporal validity durations and improving the generalizability to different text sources, rather than for a direct downstream task application.

References

Shun Abe, Masumi Shirakawa, Tatsuya Nakamura, Takahiro Hara, Kazushi Ikeda, and Keiichiro Hoashi. 2018. Predicting the occurrence of life events from user's tweet history. In *2018 IEEE 12th International Conference on Semantic Computing (ICSC)*, pages 219–226. IEEE.

Axel Almquist and Adam Jatowt. 2019. Towards content expiry date determination: Predicting validity periods of sentences. In *European Conference on Information Retrieval*, pages 86–101. Springer.

Sitaram Asur and Bernardo A Huberman. 2010. Predicting the future with social media. In *2010 IEEE/WIC/ACM international conference on web intelligence and intelligent agent technology*, volume 1, pages 492–499. IEEE.

Prajjwal Bhargava and Vincent Ng. 2022. Commonsense knowledge reasoning and generation with pre-trained language models: a survey. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 12317–12325.

Ning Bian, Xianpei Han, Le Sun, Hongyu Lin, Yaojie Lu, and Ben He. 2023. Chatgpt is a knowledgeable but inexperienced solver: An investigation of commonsense problem in large language models. *arXiv preprint arXiv:2303.16421*.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, 6.

Bibo Cai, Xiao Ding, Bowen Chen, Li Du, and Ting Liu. 2022. Mitigating reporting bias in semi-supervised temporal commonsense inference with probabilistic soft logic. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10454–10462.

Shangkun Deng, Takashi Mitsubuchi, Kei Shioda, Tatsuro Shimada, and Akito Sakurai. 2011. Combining technical analysis with sentiment analysis for stock price prediction. In *2011 IEEE ninth international conference on dependable, autonomic and secure computing*, pages 800–807. IEEE.

Chadi Helwe, Chloé Clavel, and Fabian M Suchanek. 2021. Reasoning with transformer-based models: Deep learning, but shallow reasoning. In *3rd Conference on Automated Knowledge Base Construction*.

Hidehito Honda, Rina Kagawa, and Masaru Shirasuna. 2022. On the round number bias and wisdom of crowds in different response formats for numerical estimation. *Scientific Reports*, 12(1):1–18.

Taishi Hosokawa, Adam Jatowt, Masatoshi Yoshikawa, and Kazunari Sugiyama. 2023. Temporal natural language inference: Evidence-based evaluation of temporal text validity. *Proceedings of the 45th European Conference on Information Retrieval (ECIR 2023), Springer LNCS*.

Marc W Howard. 2018. Memory as perception of the past: compressed time inmind and brain. *Trends in cognitive sciences*, 22(2):124–136.

Adam Jatowt and Ching-man Au Yeung. 2011. Extracting collective expectations about the future from large text collections. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1259–1264.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2.

Mayuko Kimura, Lis Kanashiro Pereira, and Ichiro Kobayashi. 2021. Towards a language model for temporal commonsense reasoning. In *Proceedings of the Student Research Workshop Associated with RANLP 2021*, pages 78–84.

Yashasvi Koul, Kanishk Mamgain, and Ankit Gupta. 2022. Lifetime of tweets: a statistical analysis. *Social Network Analysis and Mining*, 12(1):101.

Niels Buus Lassen, Rene Madsen, and Ravi Vatrapu. 2014. Predicting iphone sales from iphone tweets. In *2014 IEEE 18th International Enterprise Distributed Object Computing Conference*, pages 81–90. IEEE.

Pengfei Li, Hua Lu, Nattiya Kanhabua, Sha Zhao, and Gang Pan. 2018. Location inference for non-geotagged tweets in user timelines. *IEEE Transactions on Knowledge and Data Engineering*, 31(6):1150–1165.

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 preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Yafeng Lu, Robert Krüger, Dennis Thom, Feng Wang, Steffen Koch, Thomas Ertl, and Ross Maciejewski. 2014. Integrating predictive analytics and social media. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 193–202. IEEE.

Steven Lynden, Mehari Heilemariam, Kyoung-Sook Kim, Adam Jatowt, Akiyoshi Matono, Hai-Tao Yu, Xin Liu, and Yijun Duan. 2023. Commonsense temporal action knowledge (cotak) dataset. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM 2023)*.

Bill MacCartney. 2009. *Natural language inference*. Stanford University.

James Manyika. 2023. An overview of bard: an early experiment with generative ai.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849.

Abhilash Nandy, Sushovan Haldar, Subhashis Banerjee, and Sushmita Mitra. 2020. A survey on applications of siamese neural networks in computer vision. In *2020 International Conference for Emerging Technology (INCET)*, pages 1–5. IEEE.

Qiang Ning, Hao Wu, Rujun Han, Nanyun Peng, Matt Gardner, and Dan Roth. 2020. Torque: A reading comprehension dataset of temporal ordering questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1158–1172.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Lis Pereira, Fei Cheng, Masayuki Asahara, and Ichiro Kobayashi. 2021. Alice++: Adversarial training for robust and effective temporal reasoning. In *Proceedings of the 35th Pacific Asia Conference on Language, Information and Computation*, pages 373–382.

Lis Pereira, Xiaodong Liu, Fei Cheng, Masayuki Asahara, and Ichiro Kobayashi. 2020. Adversarial training for commonsense inference. In *Proceedings of the 5th Workshop on Representation Learning for NLP (RepL4NLP-2020)*, pages 55–60. Association for Computational Linguistics.

Lianhui Qin, Aditya Gupta, Shyam Upadhyay, Luheng He, Yejin Choi, and Manaal Faruqui. 2021. Timedial: Temporal commonsense reasoning in dialog. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the*

11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 7066–7076.

Guy Rosin and Kira Radinsky. 2022. Temporal attention for language models. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1498–1508.

Guy D Rosin, Ido Guy, and Kira Radinsky. 2022. Time masking for temporal language models. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 833–841.

Lingfeng Shen, Zhuoming Liu, and Xiongtao Zhou. 2020. Forecasting people's action via social media data. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 5254–5259. IEEE.

Shane Storks, Qiaozi Gao, and Joyce Y Chai. 2019a. Commonsense reasoning for natural language understanding: A survey of benchmarks, resources, and approaches. *arXiv preprint arXiv:1904.01172*, pages 1–60.

Shane Storks, Qiaozi Gao, and Joyce Y Chai. 2019b. Recent advances in natural language inference: A survey of benchmarks, resources, and approaches. *arXiv preprint arXiv:1904.01172*.

Zijun Sun, Chun Fan, Qinghong Han, Xiaofei Sun, Yuxian Meng, Fei Wu, and Jiwei Li. 2020. Self-explaining structures improve nlp models. *arXiv preprint arXiv:2012.01786*.

Hikaru Takemura and Keishi Tajima. 2012. Tweet classification based on their lifetime duration. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 2367–2370.

Qingyu Tan, Hwee Tou Ng, and Lidong Bing. 2023. Towards benchmarking and improving the temporal reasoning capability of large language models. *arXiv preprint arXiv:2306.08952*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Lav R Varshney and John Z Sun. 2013. Why do we perceive logarithmically? *Significance*, 10(1):28–31.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Jiexin Wang, Adam Jatowt, Masatoshi Yoshikawa, and Yi Cai. 2023. Bitimebert: Extending pre-trained language representations with bi-temporal information. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 812–821.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Georg Wenzel and Adam Jatowt. 2023. An overview of temporal commonsense reasoning and acquisition. *arXiv preprint arXiv:2308.00002*.

Zonglin Yang, Xinya Du, Alexander M Rush, and Claire Cardie. 2020. Improving event duration prediction via time-aware pre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3370–3378.

Changlong Yu, Hongming Zhang, Yangqiu Song, and Wilfred Ng. 2022. Cocolm: Complex commonsense enhanced language model with discourse relations. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1175–1187.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020. Reasoning about goals, steps, and temporal ordering with wikihow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4630–4639.

Ben Zhou, Daniel Khashabi, Qiang Ning, and Dan Roth. 2019. "going on a vacation" takes longer than "going for a walk": A study of temporal commonsense understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics.

Ben Zhou, Qiang Ning, Daniel Khashabi, and Dan Roth. 2020. Temporal common sense acquisition with minimal supervision. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7579–7589.

Ben Zhou, Kyle Richardson, Qiang Ning, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2021. Temporal reasoning on implicit events from distant supervision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1361–1371.

Bo Zhou, Yubo Chen, Kang Liu, Jun Zhao, Jiexin Xu, Xiaojian Jiang, and Qiuxia Li. 2022. Generating temporally-ordered event sequences via event optimal transport. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1875–1884.

A Crowdsourcing Definitions

In this section, we provide details on the crowdsourcing implementation. As noted, we use Amazon Mechanical Turk to collect crowdsourced data from participants.

A.1 Temporal Validity Duration Estimation

We assume the average layman is not familiar with the term *temporal validity*. Thus, we define the task as "determining how long the information within the tweet remains relevant after its publication", i.e., for how long the user would consider the tweet timely and relevant. We provide the option *no time-sensitive information*, which should be selected when tweets do not contain any information, when information is not expected to change over time, or when it is fully contained in the past.

The task is otherwise a relatively straightforward classification task. We split our dataset into batches of 10 samples that are grouped into a single *human intelligence task* (HIT). For each HIT, we offer a compensation of USD0.25, based on an estimated 6-9 seconds of processing time per individual statement (i.e., 60-90 seconds per HIT). Figures 10 to 13 show the crowdsourcing layout.

A.2 Follow-Up Content Generation

Compared to the temporal validity duration estimation task, the follow-up content generation task requires a much more robust understanding of the overall concept of temporal validity and the respective semantic roles of the target- and follow-up statements. Hence, we focus on providing a more detailed explanation of the task. Figures 14 to 16 show the crowdsourcing setup. The detailed instructions tab is not listed due to its length, but contains instructions that can also be found in the public repository.

Notably, we labelled the target statement as *context tweet* rather than *target tweet* in this crowdsourcing task to emphasize that participants should not alter this statement directly, as this was a problem that occurred somewhat frequently during pilot tests. This contrasts with our formal definition of TVCP, where providing context is the role of the follow-up statement.

Each HIT requires participants to provide three follow-up statements, one for each TVCP class (DEC, UNC, INC). For each HIT, we offer a compensation of USD0.35. We base our compensation on an estimated 30–40 seconds of processing time per follow-up statement (i.e., 90–120 seconds per HIT) due to the creative writing involved .

A.3 Discouraging Dishonest Activity

In initial pilot runs, we find that many submissions are the result of spam, dishonest activity, or a complete lack of task understanding, with many provided annotations being inexplicable by common sense in any possible interpretation of the statement.

To increase the quality of work on both tasks, we introduced three measures. First, we required participants to have an overall approval rate of 90% on the platform, as well as 1,000 approved HITs. Without these requirements, the amount of blatant spam (e.g., copy-pasted content) increases significantly.

Second, we devised qualification tests for both tasks. Participants had to determine the temporal validity durations for a set of sample statements to work on the temporal validity duration estimation task, and determine the correctness of follow-up statements and their updated duration labels to work on the follow-up content generation task.

Finally, we vet all participants' responses individually up to a certain threshold. For each task, we manually verify the first 20 submissions of each annotator on their quality. We provide feedback and manually adapt submissions when they are partially incorrect. If submission quality is appropriate by the time a participant reaches 20 submitted HITs, we consider them as trusted, and only spot-check every 5th submission thereafter. If submission quality does not sufficiently improve at this point, we prohibit the participant from further working on the task.

Despite these efforts, the follow-up content generation task specifically still received several low-quality submissions that had to be manually filtered out and corrected. In future work, a preferable approach may be to replace the qualification test with an unpaid qualification HIT, in which a feedback loop between participants and requesters can be established on data that will not be included in the final dataset, and participants can manually be assigned a qualification once their quality of work is sufficient.

B ChatGPT Setup

We provide the following system prompt to the ChatGPT API:

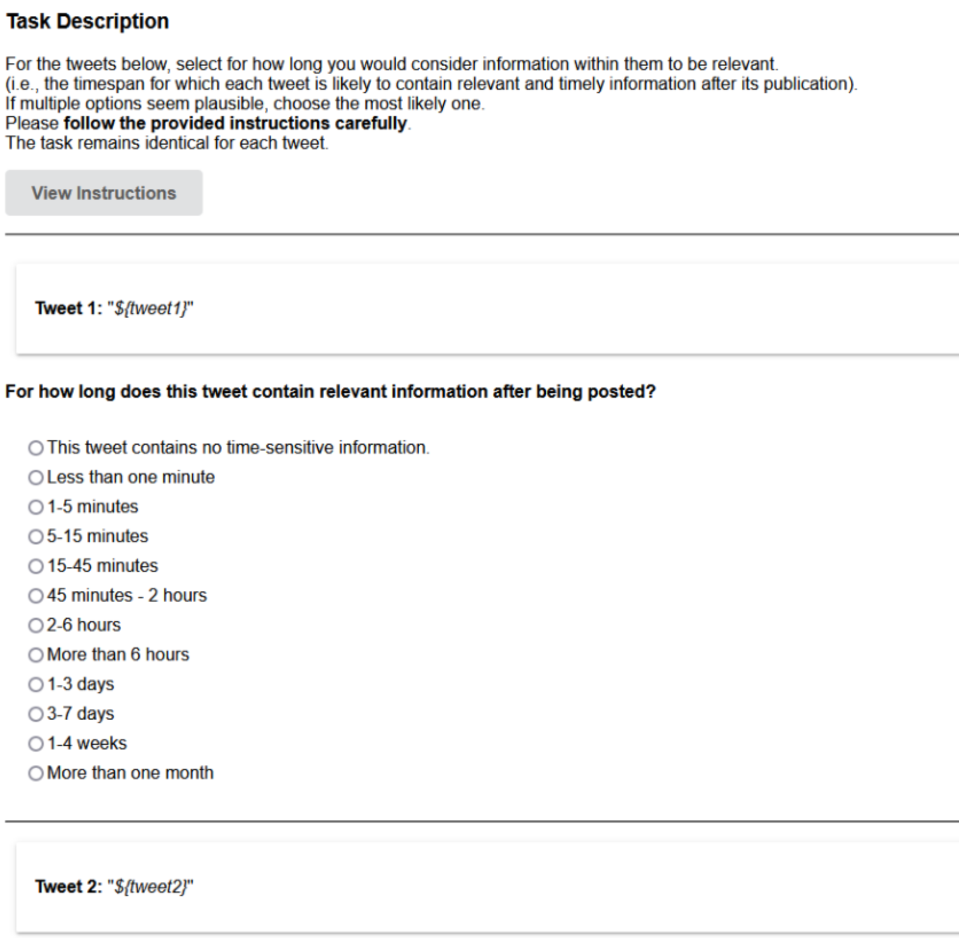

Figure 10: The interface of the temporal validity duration estimation task

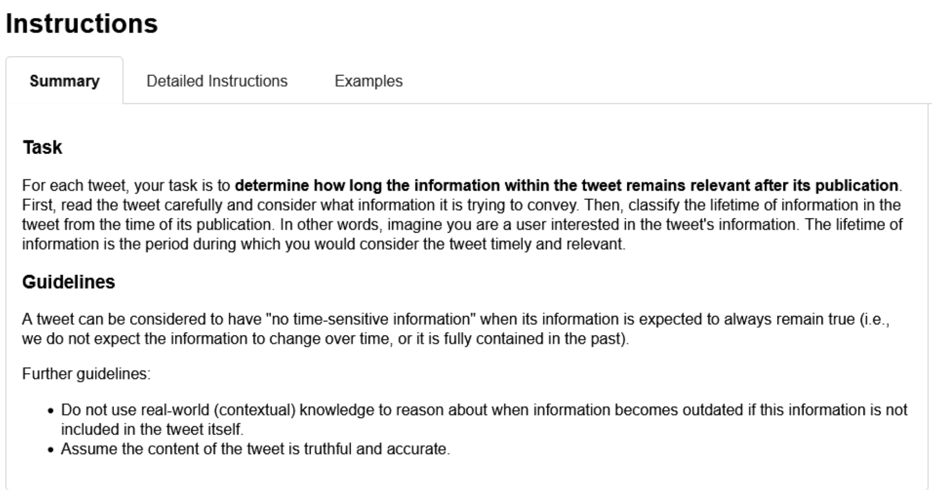

Figure 11: The summary section of the temporal validity duration estimation task guidelines

Instructions

Summary | **Detailed Instructions** | Examples

Task

The goal of this task is to gather commonsense judgments about the duration of relevance for actions and events commonly discussed on social media. For each tweet, your task is to **determine how long the information within the tweet remains relevant after its publication**. First, read the tweet carefully and consider what information it is trying to convey. Then, classify the lifetime of information in the tweet from the time of its publication. In other words, imagine you are a user interested in the tweet's information. The lifetime of information is the period during which you would consider the tweet timely and relevant. For example, a tweet like "*Check out the circus, coming to town this weekend only!*" would have a lifespan of "3-7 days" (specifically, until the end of the week). If someone were to read this tweet a few weeks after it was posted, the information would have lost its value.

Guidelines

A tweet can be considered to have "no time-sensitive information" in the following cases:

- The tweet contains information that is not expected to change over time (e.g., "*My name is Georg.*" or "*Japan lies in Asia.*").
- The tweet contains no information at all (e.g., "*Dartsssss*" or "*Endless.*").
- The tweet contains information that is fully contained in the past (e.g., "*I slept for 10 hours.*"). This **also applies** if such a statement is tied to a temporal expression (e.g., "*I slept for 10 hours yesterday.*"). In this case, despite the statement being tied to the current day due to the expression "*yesterday*", since the actual information is fully contained in the past (and the action is already fully completed), the sentence is considered to have no time-sensitive information. This is because a statement about past actions or events is considered to always remain true.

Further guidelines:

- Do not use real-world knowledge (i.e., contextual knowledge about entities or events that is not stated in the tweet itself) to reason about when information becomes outdated. For example, for the sentence "*The world cup finals are coming up.*", do not consider the actual date of the next world cup finals, but rather consider how far before the finals of any given world cup someone would be expected to post this tweet.
- Assume the content of the tweet is truthful. For example, for the sentence "*I am going to meet the queen.*", do not consider the actual likelihood of this event occurring or real-life circumstances which cause this particular event to be impossible, but instead, assume that information in the tweet holds true and that events are expected to occur.

Figure 12: The detailed description of the temporal validity duration estimation task guidelines

Instructions

Summary | Detailed Instructions | **Examples**

Good examples	**Bad examples**
Tweet: *This breakfast was pretty bad, but at least I'm going out for dinner tonight.* Classification: More than 6 hours Comment: As the user mentions breakfast, we can assume this tweet was written early in the day. Without this context, "2-6 hours" would also be acceptable.	Tweet: *This day is awful! I don't even know how it could get any worse.* Classification: 1-3 days Comment: Since the tweet is only relevant on the current day, the correct classification is "More than 6 hours". "1-3 days" should only be used as a classification when the tweet is relevant until at least the next day.
Tweet: *I hate Thursdays.* Classification: No time-sensitive information. Comment: The tweet is phrased in a way that implies it is a recurring feeling and not limited to the current week. Thus, we do not expect this sentiment to change.	Tweet: *This year all my family is getting coal and a hug.* Classification: Less than one minute Comment: The target action (giving family coal and a hug) may take less than one minute. However, unless we expect the action to take place immediately, this is not equal to the duration of relevance of the tweet.
Tweet: *I just want to finish getting all of my tattoos so badly, but I have more important things to spend money on right now.* Classification: More than one month Comment: Note the user's intent to finish getting their tattoos, which indicates that the tweet contains time-sensitive information. However, the tweet indicates that this change is not expected to occur soon.	Tweet: *Idk if i wanna go to dc today or tomorrow* Classification: No time-sensitive information Comment: Even though there is no concrete action specified, the intents of the user are focused on a specific duration. The correct classification is "1-3 days".

Figure 13: The examples section of the temporal validity duration estimation task guidelines

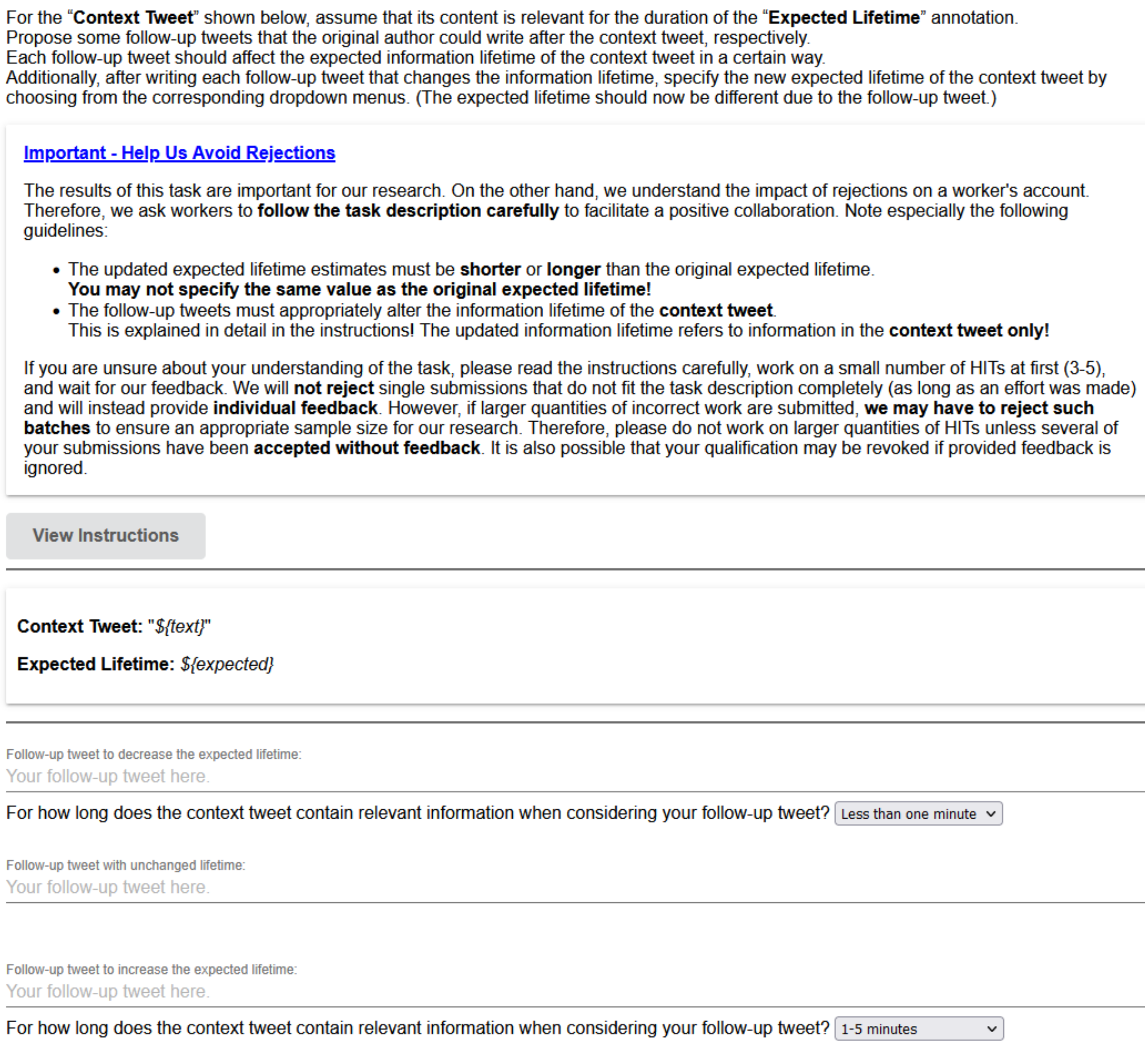

Figure 14: The interface of the follow-up content generation task

Instructions

Summary | Detailed Instructions | Examples

Task

In this crowdsourcing task, you are given a context tweet with an "expected lifetime" that indicates how long the information in the tweet will be relevant. Your task is to write three follow-up tweets:

- One where the expected lifetime of information in the context tweet **decreases**.
- One where the expected lifetime of information in the context tweet **remains unchanged**.
- One where the expected lifetime of information in the context tweet **increases**.

For the follow-up tweets that change the expected lifetime, you must also provide an updated lifetime estimate for the context tweet. Note that this new estimate is a period starting at the creation of the context tweet, **not** the follow-up tweet. Additionally, it must be a different class than the original expected lifetime (at least the adjacent shorter/longer class).

Guidelines

- Do not change the context tweet itself. Write follow-up tweets instead.
- You may not specify the same value as the original expected lifetime for your updated lifetime estimates.
- Focus on changing the lifetime of information in the context tweet.
- Give your best effort when the context tweet is unclear.
- Try to avoid using explicit temporal expressions.
- Be creative and come up with varied scenarios that change the expected information lifetime.

Possible Reasons for Rejection

We appreciate your contributions to our crowdsourcing tasks and strive to avoid rejecting work. However, in cases where the work submitted does not meet the requirements of the provided task, we may be unable to issue payment. **Work may be rejected if you submit a large number of HITs that do not follow the task description**. Most notably, some reasons for rejection may be:

- The work submitted does not adhere to the task description, especially the guidelines highlighted within the HIT interface and the instruction summary.
- The work appears to be "low-effort" (e.g., simply stating that an action will take a certain amount of time without providing further context).
- The work is written in poor English. While perfect grammar is not required, the level of English should at least match that of the context tweet.
- The provided updated lifetime estimates do not follow the task description. For instance, if the objective is to increase the lifetime of information, the work may be rejected if the updated lifetime estimate is not longer than the original estimate.

Figure 15: The summary section of the follow-up content generation task guidelines

Instructions

Summary | Detailed Instructions | **Examples**

Good examples	**Bad examples**
Context Tweet: "*Going to the gym after work today!*" **Expected Lifetime:** More than 6 hours	**Context Tweet:** "*Going to the gym after work today!*" **Expected Lifetime:** More than 6 hours
Follow-up to decrease the expected lifetime: "*Actually, I'll get a quick workout in during my lunch break at the gym next door.*" New expected lifetime: *2-6 hours* **Why?** The main information in the context tweet (going to the gym / working out) remains valid, but the action will now occur within a more immediate timeframe.	Follow-up to decrease the expected lifetime: "*I'm so sore from yesterday's workout that I can barely move. Skipping the gym today.*" New expected lifetime: *Less than one minute* **Why?** Consider the difference between an action that does not occur at all, and an action that occurs very quickly. In this example, the context tweet's information does not apply.
Follow-up with unchanged lifetime: "*I think I'll try out a new HIIT workout.*" **Why?** The follow-up tweet relates to the context tweet, but does not change the expected lifetime of information.	Follow-up with unchanged lifetime: "*I think I'll get pizza for dinner tonight.*" **Why?** In the unchanged lifetime task, there should still be some topical connection between the follow-up and the context tweet.
Follow-up to increase the expected lifetime: "*Have to work overtime today. The gym will have to wait until tomorrow.*" New expected lifetime: *1-3 days* **Why?** As the author confirms that the planned action will still take place, we consider the information lifetime in the context tweet as continuously valid until this new date.	Follow-up to increase the expected lifetime: "*The gym is closed today due to a maintenance issue. Guess I'm not going.*" New expected lifetime: *1-3 days* **Why?** A follow-up tweet cancelling plans can only be considered an appropriate follow-up when it is clear the action will still take place at a later date, which is not the case in this example.

Figure 16: The examples section of the follow-up content generation task guidelines

> "You are a language model specializing in temporal commonsense reasoning. Each prompt contains Sentence A and Sentence B. You should determine whether Sentence B changes the expected temporal validity duration of Sentence A, i.e., the duration for which the information in Sentence A is expected to be relevant to a reader.
>
> To achieve this, in your responses, first, estimate for how long the average reader may expect Sentence A to be relevant on its own. Then, consider if the information introduced in Sentence B increases or decreases this duration. Surround this explanation in triple backticks (```).
>
> After your explanation, respond with one of the three possible classes corresponding to your explanation: Decreased, Neutral, or Increased."

After this system prompt, we provide three sample responses, one for each of the classes. These sample responses are shown in Figure 17.

Similar to the crowdsourcing task setup, we use the concept of the *expected relevance duration* of the target statement (called Statement A in the CHATGPT prompt) to explain statement-level temporal validity. Additionally, instead of prompting the model to classify the sample directly, we ask it to provide an explanation for its decision based on chain-of-thought reasoning. Wei et al. (2022) show that chain-of-thought prompting significantly increases several types of reasoning capabilities, including commonsense, in LLMs. We prompt CHATGPT to first estimate the temporal validity duration of the target statement on its own. In a second step, the model should then determine if the information introduced in the follow-up statement changes this temporal validity duration. After giving its explanation, the model should respond with one of the three target classes.

Sentence A: I'm ready to go to the beach
Sentence B: I forgot all the beach towels are still in the dryer, but I'll be ready to go as soon as the dryer's done running.
Target Class: Increased
Sample Explanation: Going to the beach may take a few minutes to an hour, depending on the distance. However, if the author first needs to wait on the dryer to finish in order to retrieve their beach towels, this may take an additional 30-60 minutes.

Sentence A: taking bad thoughts out of my mind thru grinding my assignments
Sentence B: I just have to get through a short math homework assignment and memorize a few spelling words so it shouldn't take long.
Target Class: Decreased
Sample Explanation: Grinding through assignments may take several hours, depending on the number of assignments to complete. In Sentence B, the author states they only have a few short assignments remaining, so they may only take an hour or less to finish them.

Sentence A: Slide to my dm guys, come on
Sentence B: Instagram DMs are such a fun way to communicate.
Target Class: Neutral
Sample Explanation: The author encourages people to direct message them, which may be relevant for several minutes to a few hours. Sentence B does not change the duration for which Sentence A is expected to be relevant.

Figure 17: Sample items, target classes, and explanations provided to CHATGPT for few-shot reasoning

C Hyperparameters

We perform hyperparameter testing regarding dropout probability before the classification layer (0.1, 0.25, 0.5), the base learning rate (1e-2, 1e-3, 1e-4), and whether to freeze embedding layers (i.e., training only intermediary and classification layers). For both BERT and ROBERTA in the frozen and unfrozen setting, we perform grid-search over the learning rate and dropout probability. For these benchmarks, we use a predefined train-val-test split (80%/10%/10%). The remaining setup is the same as in Section 5.

Table 4 shows the three best-performing configurations for BERT and ROBERTA in the freeze and nofreeze settings, respectively, on the TRANSFORMERCLASSIFIER pipeline. Table 5 shows the same results for the SIAMESECLASSIFIER pipeline.

The most notable finding appears to be that

Model	DO	LR	#Epochs	EM
BERT-nofreeze	0.25	1e-4	5	0.613
BERT-nofreeze	0.10	1e-4	6	0.548
BERT-nofreeze	0.50	1e-4	4	0.548
BERT	0.25	1e-4	17	0.321
BERT	0.10	1e-4	8	0.315
BERT	0.10	1e-3	10	0.304
RoBERTa	0.25	1e-3	14	0.262
RoBERTa	0.10	1e-4	16	0.256
RoBERTa	0.50	1e-3	15	0.238
RoBERTa-nofreeze	0.25	1e-3	1	0.000
RoBERTa-nofreeze	0.50	1e-3	1	0.000
RoBERTa-nofreeze	0.10	1e-4	1	0.000

Table 4: Best three models for each of the proposed configurations in the TransformerClassifier pipeline

Model	DO	LR	#Epoch	EM
BERT-nofreeze	0.25	1e-4	7	0.589
BERT-nofreeze	0.10	1e-4	4	0.577
BERT-nofreeze	0.50	1e-4	2	0.565
RoBERTa	0.10	1e-4	21	0.548
RoBERTa	0.50	1e-4	13	0.518
RoBERTa	0.25	1e-4	17	0.512
BERT	0.50	1e-4	9	0.387
BERT	0.25	1e-3	8	0.357
BERT	0.25	1e-4	5	0.339
RoBERTa-nofreeze	0.25	1e-3	1	0.000
RoBERTa-nofreeze	0.50	1e-3	1	0.000
RoBERTa-nofreeze	0.10	1e-4	1	0.000

Table 5: Best three models for each of the proposed configurations in the SiameseClassifier pipeline

RoBERTa gets stuck in a false minimum of predicting a constant class when embedding layers are unfrozen, leading to an accuracy of 0.33 and an EM of 0. Hence, we freeze embedding layers for these model types in our main evaluation. As noted in Section 5, a possible reason for this behaviour could be differences in the embeddings contained within BERT's [CLS] and RoBERTa's <s> token.