Modeling Human Mental States with an Entity-based Narrative Graph

I-Ta Lee, Maria Leonor Pacheco, Dan Goldwasser
Department of Computer Science
Purdue University
West Lafayette, IN, USA
{lee2226, pachecog, dgoldwas}@purdue.edu

Abstract

Understanding narrative text requires capturing characters' motivations, goals, and mental states. This paper proposes an Entity-based Narrative Graph (ENG) to model the internal-states of characters in a story. We explicitly model entities, their interactions and the context in which they appear, and learn rich representations for them. We experiment with different task-adaptive pre-training objectives, in-domain training, and symbolic inference to capture dependencies between different decisions in the output space. We evaluate our model on two narrative understanding tasks: predicting character mental states, and desire fulfillment, and conduct a qualitative analysis.

1 Introduction

Understanding narrative text requires modeling the motivations, goals and internal states of the characters described in it. These elements can help explain intentional behavior and capture causal connections between the characters' actions and their goals. While this is straightforward for humans, machine readers often struggle as a correct analysis relies on making long range common-sense inferences over the narrative text. Providing the appropriate narrative representation for making such inferences is therefore a key component. In this paper, we suggest a novel narrative representation model and evaluate it on two narrative understanding tasks, analyzing the characters' mental states and motivations (Abdul-Mageed and Ungar, 2017; Rashkin et al., 2018; Chen et al., 2020), and desire fulfillment (Chaturvedi et al., 2016; Rahimtoroghi et al., 2017).

We follow the observation that narrative understanding requires an expressive representation capturing the context in which events appear and the interactions between characters' states. To clarify, consider the short story in Fig. 1. The desire expression appears early in the story and provides the context explaining the protagonist's actions. Evaluating the fulfilment status of this expression,

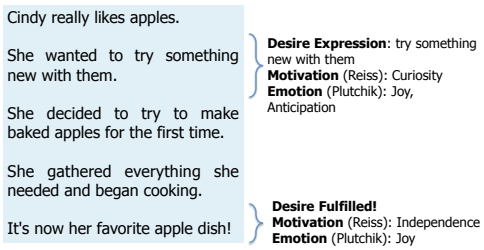


Figure 1: Narrative Example

which tends to appear towards the end of the story, requires models that can reason over the desire expression (*"trying something new"*), its target (*"apples"*) and the outcome of the protagonist's actions (*"it's now her favorite apple dish!"*). Capturing the interaction between the *motivation* underlying the desire expression (in Fig. 1, CURIOSITY) and the *emotions* (in Fig. 1, ANTICIPATION) likely to be invoked by the motivation can help ensure the consistency of this analysis and improve its quality.

To meet this challenge, we suggest a graph-contextualized representation for entity states. Similar to contextualized word representations (Peters et al., 2018; Devlin et al., 2019), we suggest learning an entity-based representation which captures the narrative it is a part of. For example, in *"She decided to try to make baked apples for the first time"* the mental state of "she" would be represented differently given a different context, such as a different motivation for the action (*"Her mother asked her to make an apple dish for a dinner party"*). In this case, the contextualized representation would capture the different emotion associated with it (e.g., FEAR of disappointing her mother). Unlike contextualized word embeddings, *entity-based* contextualization needs to consider, at least, two levels of context: local text context and distant event context, which require more complicated modeling techniques to capture event semantics. Moreover,

Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 4916–4926
June 6–11, 2021.

the context of event relationships can spread over a long narrative, exceeding maximum sequence length limitation in modern contextualized word embedding models such as BERT (Devlin et al., 2019).

In this paper, we propose an Entity-based Narrative Graph (ENG) representation of the text. Unlike other graph-based narrative representations (Lehnert, 1981; Goyal et al., 2010; Elson, 2012) which require intensive human annotation, we design our models around low-cost supervision sources and shift the focus from symbolic graph representations of nuanced information to their learned embedding. In ENG, each node is associated with an entity-event pair, representing an entity mention that is involved in an event. Edges represent observed relations between entities or events. We adapt the definition of event relationships introduced in Lee et al. (2020) to our entity-event scenario. For entity relationships, the **CNext** relationship connects two coreferent entity nodes. For event relationships, the **Next** relationship captures the sequential order of events as they appear in the text, and six discourse relation types from the Penn Discourse Tree Bank (PDTB) (Prasad et al., 2007) are used. These include **Before, After, Sync., Contrast, Reason** and **Result**. Note that these are extracted in a weakly supervised manner, without expensive human annotations.

To contextualize the entity embeddings over ENG, we apply a Relational Graph Convolution Network (R-GCN) (Schlichtkrull et al., 2018), a relational variant of the Graph Convolution Network architecture (GCN) (Kipf and Welling, 2016). R-GCNs create contextualized node representations by considering the graph structure through graph convolutions and learn a composition function. This architecture allows us to take into account the narrative structure and the different discourse relations connecting the entity-event nodes.

To further enhance our model, we investigate three possible pre-training paradigms: whole-word-masking, node prediction, and link prediction. All of them are constructed by automatically extracting noisy supervision and pre-training on a large-scale corpus. We show that choosing the right pre-training strategy can lead to significant performance enhancements in downstream tasks. For example, automatically extracting sentiment for entities can impact downstream emotion predictions. Finally, we explore the use of a symbolic inference layer to model relationships in the output space, and show that we can obtain additional gains in the downstream tasks that have strong correlation in the output space.

The evaluated downstream tasks include two challenging narrative analysis tasks, predicting characters' psychological states (Rashkin et al., 2018) and desire fulfilment (Rahimtoroghi et al., 2017). Results show that our model can outperform competitive transformer-based representations of the narrative text, suggesting that explicitly modeling the relational structure of entities and events is beneficial. Our code and trained models are publicly available[1].

2 Related Work

Tracking entities and modeling their properties has proven successful in a wide range of tasks, including language modeling (Ji et al., 2017), question answering (Henaff et al., 2017) and text generation (Bosselut et al., 2018). In an effort to model complex story dynamics in text, Rashkin et al. (2018) released a dataset for tracking the emotional reactions of characters in stories. In their dataset, each character mention is annotated with three types of mental state descriptors: Maslow's "hierarchy of needs" (Maslow, 1943), Reiss' "basic motives" (Reiss, 2004), that provide a more informative range of motivations, and Plutchik's "wheel of emotions" (Plutchik, 1980), comprised of eight basic emotional dimensions (e.g. joy, sadness, etc). In their paper, they showed that neural models with explicit or latent entity representations achieve promising results on this task. Paul and Frank (2019) approached this task by extracting multi-hop relational paths from ConceptNet, while Gaonkar et al. (2020) leveraged semantics of the emotional states by embedding their textual description and modeling the co-relation between different entity states. Rahimtoroghi et al. (2017) introduced a dataset for the task of desire fulfillment. They identified desire expressions in first-person narratives and annotated their fulfillment status. They showed that models that capture the flow of the narrative perform well on this task.

Representing the narrative flow of stories using graph structures and multi-relational embeddings has been studied in the context of script learning (Li et al., 2018; Lee and Goldwasser, 2019; Lee et al.,

[1] https://github.com/doug919/entity_based_narrative_graph

2020). In these cases, the nodes represent predicate-centric events, and entity mentions are added as context to the events. In this paper, we use an entity-centric narrative graph, where nodes are defined by entity mentions and their textual context. We encode the textual information in the nodes using pre-trained language models (Devlin et al., 2019; Liu et al., 2019), and the graph structure with a relational graph neural network (Schlichtkrull et al., 2018). To learn the representation, we incorporate a task-adaptive pre-training phase. Gururangan et al. (2020) showed that further specializing large pre-trained language models to domains and tasks within those domains is effective.

3 Entity-based Narrative Graph

3.1 Framework Overview

Many NLU applications require understanding entity states in order to make sophisticated inferences (Sap et al., 2018; Bosselut et al., 2019; Rashkin et al., 2018), and the entity states are highly related to the event the entity involves in. In this work, we propose a learning framework that aims at modeling entities' internal states, and their interactions to other entities' internal states through events. We include task-adaptive pre-training (TAPT) and downstream task training to train an entity-based narrative graph (ENG), a graph neural model designed to capture implicit states and interactions between entities. We extend the narrative graph proposed by Lee et al. (2020), which models event relationships, and instead of learning node representations for events, we focus on entity mentions that are involved in events. This change is motivated by the high-demand of NLU applications that require understanding entity mentions' states in order to make sophisticated inference.

Our framework consists of four main components: Node Encoder, Graph Encoder, Learning Objectives, and Symbolic Inference, outlined in Figure 2. The node encoder is a function used to extract event information about the target entity mention corresponding to the local node representation. The graph encoder uses a graph neural network to contextualize node representations with entity-events in the same document, generating entity-context-aware representations. The learning objectives use this representation for several learning tasks, such as node classification, link prediction, and document classification. Finally, we include a symbolic inference procedure to capture dependencies between output decisions.

We introduce a training pipeline, containing pre-training and downstream training, following recent evidence suggesting that task-adaptive pre-training is potentially useful for many NLU tasks (Gururangan et al., 2020). We experiment with three pre-training setups, including the common whole-word-masking pre-training (Liu et al., 2019), and two newly proposed unsupervised pre-training objectives based on ENG. We then evaluate two downstream tasks: StoryCommonsense (Rashkin et al., 2018) and DesireDB (Rahimtoroghi et al., 2017). StoryCommonsense aims at predicting three sets of mental states based on psychological theories (Maslow, 1943; Reiss, 2004; Plutchik, 1980), while DesireDB's goal is to identify whether a target desire is satisfied or not. Solving these tasks requires understanding entities' mental states and their interactions.

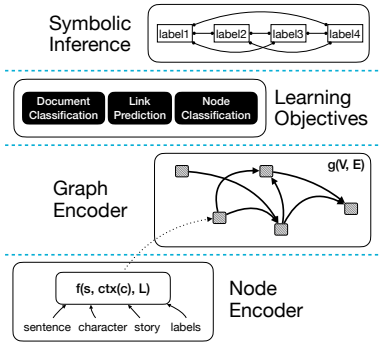


Figure 2: Overview of the ENG framework.

3.2 Node Encoder

Each node in our graph captures the local context of a specific entity mention (or character mention), and how the entity mentions are extracted is related to extracting their edges, which will be described in Sec. 3.3. Following Gaonkar et al. (2020), we format the input information to be fed into a pre-trained language model. For a given character c and sentence s, the inputs to the node encoder consist of three components $(s, ctx(c), L)$, where s is the sentence in which c appears, $ctx(c)$ is the context of c (all the sentences that the character appears in), and L is a label sentence. The label sentence is an artificial sentence of the form "[entity name] is [label 1], [label 2], ..., [label k]." The k labels correspond to the target labels in the downstream task. For example, in StoryCommonsense, the Plutchik state prediction task has eight labels characteriz-

ing human emotions, such as *joy*, *trust*, and *anger*. Gaonkar et al. (2020) show that self-attention is an effective way to let the model take label semantics into account, and improve performance[2].

Our best model uses RoBERTa (Liu et al., 2019), a highly-optimized version of BERT (Devlin et al., 2019), to encode nodes. We convert the node input $(s, ctx(c), L)$ to RoBERTa's two-sentence input format by treating s as the first sentence, and the concatenation of $ctx(c)$ and L as the second sentence. After forward propagation, we take the pooled sentence representation (i.e., <*s*>for RoBERTa, *CLS* for BERT), as the node representation v. This is formulated as $v = f_{roberta}(s, ctx(c), L)$.

3.3 Graph Encoder

The ENG is defined as $ENG = (V, E)$, where V is the set of encoded nodes in a document and E is the set of edges capturing relationships between nodes. Each edge $e \in E$ is a triplet (v_1, r, v_2), where $v_1, v_2 \in V$ and r is an edge type ($r \in R$). Following Lee et al. (2020), we use eight relation types ($|R| = 8$) that have been shown to be useful for modeling narratives. NEXT denotes if two nodes appear in neighboring sentences. CNEXT expresses the next occurrence of a specific entity following its co-reference chain. Six discourse relation types, used by Lee et al. (2020) and defined in Penn Discourse Tree Bank (PDTB) (Prasad et al., 2007), are also used in this work, including BEFORE, AFTER, SYNC., CONTRAST, REASON, RESULT. Their corresponding definition in PDTB and can be found in Table 1. Following Lee et al. (2020), we use the Stanford CoreNLP pipeline[3] (Manning et al., 2014) to obtain co-reference links and dependency trees. We use them as heuristics to extract the above relations and identify entities for TAPT[4]. Details of this procedure can be found in (Lee et al., 2020). Note that although we share the same relation definitions, our nodes are defined over entities, instead of events.

For encoding the graph, we use a Relational Graph Convolution Network (R-GCN) (Schlichtkrull et al., 2018), which is designed for Knowledge Base Completion. This architecture is capable of modeling typed edges and is resilient to noise. R-GCN is defined as:

$$h_i^{l+1} = ReLU\left(\sum_{r \in R}\sum_{u \in U_r(v_i)} \frac{1}{z_{i,r}} W_r^l h_u^l\right), \quad (1)$$

where h_i^l is the hidden representation for the i-th node at layer l and $h_i^0 = v_i$ (output of the node encoder); $U_r(v_i)$ represents v_i's neighboring nodes connected by the relation type r; $z_{i,r}$ is for normalization; and W_r^l represents trainable parameters.

Our implementation of R-GCN propagates messages between entity nodes, emulating the interactions between their psychological states, and thus enriching node representations with context. Note that our framework is flexible, and alternative node and graph encoders could be used.

Abbrev.	PDTB	Distr.
NEXT	–	50%
CNEXT	–	20%
BEFORE	Temporal.Async.Precedence	5%
AFTER	Temporal.Async.Succession	5%
SYNC.	Temporal.Synchrony	5%
CONTRAST	Comparison.Contrast	5%
REASON	Contingency.Cause.Reason	5%
RESULT	Contingency.Cause.Result	5%

Table 1: Alignment between PDTB relations and the abbreviations used in this paper. The third column in the sampling distribution.

3.4 Output Layers and Learning Objectives

We explore three learning problem types.

Node Classification For node classification, we use the contextualized node embeddings coming from the graph encoder, and plug in a k-layer feed-forward neural network on top ($k = 2$ in our case). The learning objectives could be either multi-class or multi-label. For multi-class classification, we use the weighted cross-entropy loss (CE). For multi-label classification, we use the binary cross-entropy (BCE) loss for each label[5]:

$$CE = -\frac{1}{N}\sum_{i=1}^{N} \alpha_i y_i \log(S(g(f(x_i)))), \quad (2)$$

where $S(.)$ is the Softmax function, $f(.)$ is the graph encoder, $g(.)$ is the node encoder, x_i is the input including the target node i ($(s, ctx(c), L)$) and all other nodes in the same document (or ENG), y_i is the label, and α_i is the example weight based on the label distribution of the training set..

[2]Note that all candidate labels are appended to every example, without denoting which one is the right answer. Our preliminary experiments confirm that taking label semantics into account improves performance

[3]Stanford CoreNLP v4.0 with default annotators.

[4]For StoryCommonsense, since the entity names are annotated, we simply use them.

[5]We tried weighted an unweighted BCE, and selected the unweighted one for our final model.

Link Prediction This objective tries to recover missing links in a given ENG. We sample a small portion of edges (20% in our case) as positive examples, based on the relation type distribution given in Table 1, taken from the training set. To obtain negative examples, we corrupt the positive examples by replacing one component of the edge triplet with a sampled component so that the resulting triplet does not exist in the original graph. For example, given a positive edge (e_1, r, e_2), we can create negative edges: (e'_1, r, e_2), (e_1, r', e_2), or (e_1, r, e'_2). Following Schlichtkrull et al. (2018), we score each edge sample with DistMult (Chang et al., 2014):

$$D(i, r, j) = h_i^T W_r h_j, \quad (3)$$

where W_r is a relation-specific trainable matrix (non-diagonal) and h_i and h_j are node embeddings coming from the graph encoder. A higher score indicates that the edge is more likely to be active. To learn this, we reward positive samples and penalize negative ones, using an adapted CE loss:

$$L = -\frac{1}{|T|} \sum_{(i,r,j,y) \in T} y \log(\sigma(\epsilon_r D(i, r, j))) + (1 - y) \log(1 - \sigma(\epsilon_r D(i, r, j))), \quad (4)$$

T is the sampled edges set, $y = \{0, 1\}$, $\sigma(.)$ is the Sigmoid function, and ϵ_r is the edge type weight, based on the edge sampling rate in Table 1.

Document Classification For document classifications, such as DesireDB, we aggregate the node representations from the entire ENG to form a single representation. To leverage the relative importance of each node, we add a self-attention layer on top of the graph nodes. We calculate the attention weights by attending on the query embedding (in DesireDB, this is the sentence embedding for the desire expression).

$$a_i = ReLU(W_a[h_i; h_t] + b_a)$$
$$z_i = exp(a_i)$$
$$\alpha_i = \frac{z_i}{\sum_k z_k} \quad ; \quad h_d = \sum_i \alpha_i h_i \quad (5)$$

where h_i is the i-th node representation, h_t is the query embedding, W_a and b_a are trainable parameters, and h_d is the final document representation. We then feed h_d to a two-hidden-layer classifier to make predictions. We use the loss function specified in Eq. 2.

3.5 Task-Adaptive Pre-training

Recent studies demonstrate that downstream tasks performance can be improved by performing self-supervised pre-training on the text of the target domain (Gururangan et al., 2020), called Task-Adaptive Pre-Training (TAPT). To investigate whether different TAPT objectives can provide different insights for downstream tasks, we apply three possible pre-training paradigms and compare them on StoryCommonsense. We focus on StoryCommonsense given that the dataset was created by annotating characters' mental states on a subset of RocStories (Mostafazadeh et al., 2016), a corpus with 90K short common-sense stories. This provides us with a large unlabeled resource for investigating different pre-training methods. We run TAPT on all the RocStories text[6]. We use the learning parameters suggested by Gururangan et al. (2020) and explore the following strategies:

Whole-Word Masking: Randomly masks a subset of words and asks the model to recover them from their context (Radford et al., 2019; Liu et al., 2019). We perform this task over RoBERTa, initialized with *roberta-base*.

ENG Link Prediction: Weakly-supervised TAPT over the ENG. The setup follows Sec. 3.4 (Link Prediction) to learn a model that can recover missing edges in the ENG.

ENG Node Sentiment Classification: Performs weakly-supervised sentiment TAPT. We use the Vader sentiment analysis (Hutto and Gilbert, 2014) tool to annotate the sentiment polarity for each node in the ENG, based on its sentence. The setup follows Sec. 3.4 (Node Classification).

3.6 Symbolic Inference

In addition to modeling the narrative structure in the embedding space, we add a symbolic inference procedure to capture structural dependencies in the output space for the StoryCommonsense task. To model these dependencies, we use DRaiL (Pacheco and Goldwasser, 2021), a neural-symbolic framework that allows us to define probabilistic logical rules on top of neural network potentials.

Decisions in DRaiL are modeled using rules, which can be weighted (i.e., soft constraints), or unweighted (i.e., hard constraints). Rules are formatted as horn clauses: A $\Rightarrow$ B, where A is a conjunction of observations and predicted values, and

[6]Not including the validation and testing sets of Story Cloze Test

B is the output to be predicted. Each weighted rule is associated with a neural architecture, which is used as a scoring function to obtain the rule weight. The collection of rules represents the global decision, and the solution is obtained by performing MAP inference. Given that rules are written as horn clauses, they can be expressed as linear inequalities corresponding to their disjunctive form, and thus MAP inference is defined as a linear program.

In DRaiL, parameters are trained using the structured hinge loss. This way, all neural parameters are updated to optimize the global objective. Additional details can be found in (Pacheco and Goldwasser, 2021). To score weighted rules, we used feed-forward networks over the node embeddings obtained by the objectives outlined in Sec. 3.4 and 3.5, without back-propagating to the full graph. We model the following rules:

Weighted rules We score each state, as well as *state transitions* to capture the progression in a character's mental state throughout the story.

$$\texttt{Entity}(\texttt{e}_\texttt{i}) \Rightarrow \texttt{State}(\texttt{e}_\texttt{i}, \texttt{l}_\texttt{i})$$
$$\texttt{State}(\texttt{e}_\texttt{i}, \texttt{l}_\texttt{i}) \wedge \texttt{HasNext}(\texttt{e}_\texttt{i}, \texttt{e}_\texttt{j}) \Rightarrow \texttt{State}(\texttt{e}_\texttt{j}, \texttt{l}_\texttt{j})$$

where e_i and e_j are two different mentions of the same character, and `HasNext` is a relation between consecutive sentences. `State` can be either `Maslow`, `Reiss` or `Plutchik`.

Unweighted rules There is a dependency between Maslow's "hierarchy of needs' and Reiss "basic motives" (Rashkin et al., 2018). We introduce logical constraints to disallow mismatches in the Maslow and Reiss prediction for a given mention e_i. In addition to this, we model positive and negative sentiment correlations between Plutchik labels. To do this, we group labels into positive (e.g. joy, trust), and negative (e.g. fear, sadness). We refer to this set of rules as *inter-label dependencies*.

$$\texttt{Maslow}(\texttt{e}_\texttt{i}, \texttt{m}_\texttt{i}) \wedge \neg\texttt{Align}(\texttt{m}_\texttt{i}, \texttt{r}_\texttt{i}) \Rightarrow \neg\texttt{Reiss}(\texttt{e}_\texttt{i}, \texttt{r}_\texttt{i})$$
$$\texttt{Reiss}(\texttt{e}_\texttt{i}, \texttt{r}_\texttt{i}) \wedge \neg\texttt{Align}(\texttt{m}_\texttt{i}, \texttt{r}_\texttt{i}) \Rightarrow \neg\texttt{Maslow}(\texttt{e}_\texttt{i}, \texttt{m}_\texttt{i})$$
$$\texttt{Plut}(\texttt{e}_\texttt{i}, \texttt{p}_\texttt{i}) \wedge \texttt{Pos}(\texttt{p}_\texttt{i}) \wedge \neg\texttt{Pos}(\texttt{p}_\texttt{j}) \Rightarrow \neg\texttt{Plut}(\texttt{e}_\texttt{i}, \texttt{p}_\texttt{j})$$

Given that the DesireDB task requires a single prediction for each narrative graph, we do not employ symbolic inference for this task.

4 Evaluation

Our evaluation includes two downstream tasks and a qualitative analysis. We report the results for different TAPT schemes and symbolic inference on StoryCommonsense. For the qualitative analysis, we visualize and compare the contextualized graph embeddings and contextualized word embeddings.

4.1 Data and Experiment Settings

For TAPT, we use RocStories, as it has a decent amount of documents (90K after excluding the validation and testing sets) that share the text style of StoryCommonsense. For all tasks, we use the train/dev/test splits used in previous work.

All the RoBERTa models used in this paper are initialized with *roberta-base*, and the BERT models with *bert-base-uncased*. The maximum sequence length for the language models is 160. If the input sequence exceeds this number, we will keep the label sentence untouched and cut down the main sentence. For large ENGs, such as long narratives in DesireDB, we set the maximum number of nodes to 60; all the hidden layer have 128 hidden units; and the number of layers for R-GCN is 2. For learning parameters in TAPT, we set the batch size to 256 through gradient accumulations; the optimizer is Adam (Kingma and Ba, 2014) with an initial learning rate of $\texttt{1e}-\texttt{4}$, $\epsilon = \texttt{1e}-\texttt{6}$, $\beta = (0.9, 0.98)$, weight decay 0.01, and warm-up proportion 0.06. We run TAPT for 100 epochs. For the downstream tasks, we conduct a grid search of Adam's initial learning rate from $\{\texttt{2e}-\texttt{3}, \texttt{2e}-\texttt{4}, \texttt{2e}-\texttt{5}, \texttt{2e}-\texttt{6}\}$, 5000 warm-up steps, and stop patience of 10. Model selection is done on the validation set. We report results for the best model. For learning the potentials for symbolic inference with DRaiL (Pacheco and Goldwasser, 2021), we use local normalization with a learning rate of $\texttt{1e}-\texttt{3}$, and represent neural potentials using 2-layer Feed-Forward Networks over the ENG node embeddings. All hidden layers consist of 128 units. The parameters are learned using SGD with a patience of 5, tested against the validation set. For more details, refer to (Pacheco and Goldwasser, 2021). Note that while it would be possible to back-propagate to the whole graph, this is a computationally expensive procedure. We leave this exploration for future work.

4.2 Task: StoryCommonsense

StoryCommonsense consists of three subtasks: Maslow, Reiss, and Plutchik, introduced in Sec. 2. Each subtask is a multi-label classification task, where the input is a sentence-character pair in a given story, and the output is a set of mental state labels. Each story was annotated by three annota-

Group	Models	Maslow			Reiss			Plutchik		
		Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
G1	**RANDOM**	7.45	49.99	12.96	1.76	50.02	3.40	10.35	50.00	17.15
	TF-IDF	29.79	34.56	32.00	20.55	24.81	22.48	22.71	25.24	23.91
	GLOVE	27.02	37.00	31.23	16.99	26.08	20.58	19.47	46.65	27.48
	LSTM	30.34	40.12	34.55	21.38	28.70	24.51	25.31	33.44	28.81
	CNN	29.30	44.18	35.23	17.87	37.52	24.21	24.47	38.87	30.04
	REN	26.85	44.78	33.57	16.73	26.55	20.53	25.30	37.30	30.15
	NPN	26.60	39.17	31.69	15.75	20.34	17.75	24.33	40.10	30.29
G2	**SA-ELMo***	34.91	32.16	33.48	21.23	16.53	18.59	47.33	40.86	43.86
	SA-RBERT*	43.58	30.03	35.55	24.75	18.00	20.84	46.51	45.45	45.97
	LC-BERT*	43.05	41.31	42.16	29.46	28.67	29.06	49.36	52.09	50.69
	LC-RBERT*	43.25	47.17	45.13	39.62	29.75	33.98	47.87	53.41	50.49
G3	**ENG**	43.87	51.13	47.22	37.66	36.20	36.92	48.96	56.07	52.27
	ENG+Mask	44.27	53.54	**48.47**	39.29	33.93	36.41	49.64	56.93	**53.03**
	ENG+Link	43.47	52.80	47.68	37.17	37.18	**37.18**	50.62	54.48	52.48
	ENG+Sent	45.29	50.89	47.93	36.69	36.14	36.41	49.48	57.12	**53.03**
G4	**ENG+IL**	40.90	58.03	47.98	31.67	41.19	35.81	49.93	74.95	59.93
	ENG+IL+ST	40.47	58.43	47.82	31.80	40.58	35.66	51.19	72.60	**60.04**

Table 2: Results for the StoryCommonsense task, including three multi-label tasks (Maslow, Reiss, and Plutchik), for predicting human's mental states of motivations or emotions. The star sign indicates that the result is from our re-implemented version of previous baselines.

tors and the final labels were determined through a majority vote. For Maslow and Reiss, the vote is count-based, i.e., if two out of three annotators flag a label, then it is an active label. For Plutchik, the vote is rating-based, where each label has an annotated rating, ranging from $\{0, 5\}$. If the averaged rating is larger or equal to 2, then it is an active label. This is the set-up given in the original paper (Rashkin et al., 2018). Some papers (Gaonkar et al., 2020) report results using only the count-based majority vote, resulting in scores that are not comparable to ours. Therefore, we re-implement two recent strong models proposed for this task. The Label Correlation model (LC (Gaonkar et al., 2020)) applies label semantics as input and model output space using a learned correlation matrix. The Self-Attention model (SA (Paul and Frank, 2019)) utilize attentions over multi-hop knowledge paths extracted from external corpus. We evaluate them under the same set of hyper-parameters and model selection strategies as our models.

We briefly explain all the baselines, as well as our model variants shown in Table 2. The first group (G1) are the baselines proposed in the task paper. **TF-IDF** uses TF-IDF features, trained on RocStories, to represent the target sentence s and character context $ctx(c)$, and uses a Feed-Forward Net (FFN) classifier; **GloVe** encodes the sentences with the pretrained GloVe embeddings and uses a FFN; **CNN** (Kim, 2014) replaces the FFN with a Convolutional Neural Network; **LSTM** is a two-layer bi-directional LSTM; **REN** (Henaff et al., 2017) is a recurrent entity network that learns to encode information for memory cells; and **NPN** (Bosselut et al., 2018) is an **REN** variant that includes a neural process network.

The second group (G2) of baselines are based on two recent publications–**LC** and **SA**–that showed strong performance on this task. We re-implement them and run the evaluation under the same setting as our proposed models. They originally use BERT and ELMo, respectively. To provide a fair comparison, we also train a RoBERTa variant for them (LC-RBERT and SA-RBERT). Note that the original paper of SA (Paul and Frank, 2019) reports an F1 of 59.81 on Maslow and 35.41 on Reiss, while LC (Gaonkar et al., 2020) reports 65.88 on Plutchik. However, these results are not directly comparable to ours. The discrepancy arises mainly from two points: (1) The rating-based voting, described in Sec. 4.2, is not properly applied, and (2) We do not optimize the hyper-parameter search space in our setting, given the relatively expensive pre-training. Our re-implemented versions give a better foundation for a fair comparison.

The third (G3) and fourth (G4) groups are our model variants. **ENG** is the model without TAPT; **ENG+Mask**, **ENG+Link**, and **ENG+Sent** are the models with Whole-Word-Masking (WM), Link Prediction (LP), and Node Sentiment (NS) TAPT,

respectively. In the last group, **ENG(Best) + IL** and **ENG(Best) + IL + ST** are based on our best ENG model with TAPT and adding inter-label dependencies (IL) and state transitions (ST) using symbolic inference, described in Sec. 3.6.

Table 2 reports all the results. We can see that Group 2 generally performs better than Group 1 on all three subtasks, suggesting that our implementation is reasonable. Even without TAPT, **ENG** outperforms all baselines, rendering $2-3\%$ absolute F1-score improvement. With TAPT, the performance is further strengthened. Moreover, we find that different TAPT tasks offer different levels of improvement for each subtask. The WM helps the most in Maslow and Plutchik, while the LP and NS excel in Reiss and Plutchik, respectively. This means that different TAPTs embed different information needed for solving the subtask. For example, the ability to add potential edges can be key to do motivation reasoning (Reiss), while identifying sentiment polarities (NS) can help in emotion analysis (Plutchik). This observation suggests a direction of connecting different related tasks in a joint pipeline. We leave this for future work.

Lastly, we evaluate the impact of symbolic inference. We perform joint inference over the rules defined in Sec. 3.6. On Table 2, we can appreciate the advantage of modeling these dependencies for predicting Plutchik labels. However, the same is not true for the other two subtasks, where symbolic inference increases recall at the expense of precision, resulting in no F1 improvement. Note that labels for Maslow and Reiss are sparser, accounting for 55% and 42% of the nodes, respectively. In contrast, Plutchik labels are present in 68% of the nodes.

4.3 Task: DesireDB

DesireDB (Rahimtoroghi et al., 2017) is the task of predicting whether a desire expression is fulfilled or not, given its prior and posterior context. It requires aggregating information from multiple parts of the document. If a target desire is "I want to be rich", and the character's mental changed from "sad" to "happy" along the text, we can infer that their desire is likely to be fulfilled.

We use the baseline systems described in (Rahimtoroghi et al., 2017), based on SkipThought (ST) and Logistic Regression (LR), with manually engineered lexical and discourse features. We train a stronger baseline by encoding the prior and posterior context, as well as the desire expression, using BERT. Then, we add an attention layer (Eq. 5) for the two contexts over the desire expression. The resulting three representations (the weighted prior and posterior representations, and the desire representation) are then concatenated. For ENG, we add an attention layer over the nodes to form the ENG document representation. We compare BERT and BERT+ENG document representations by feeding each of them into a two-layer FFN for classification, as described in Sec. 3.4 (Doc. Classification).

Table 3 shows the result. The BERT baseline outperforms other baselines with a large gap, 4.27% absolute increase in the averaged F1-score. Furthermore, BERT+ENG forms a better document summary for the target desire, which further increase another absolute 3.23% on the avg. F1-score. These results illustrate that ENG can be used in various settings for modeling entity information.

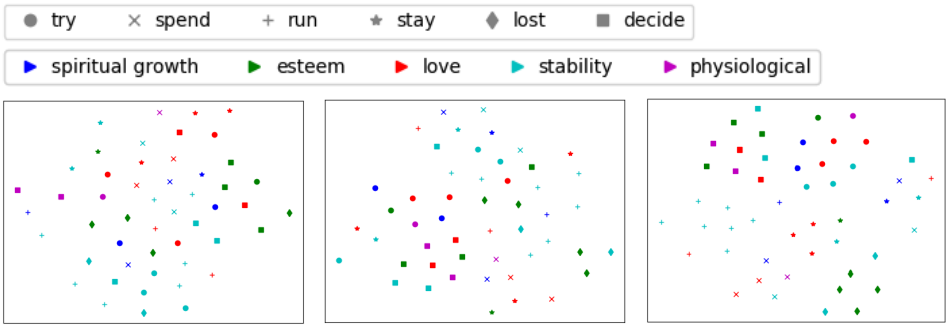
4.4 Qualitative Analysis

We conduct a qualitative analysis by measuring and visualizing distances between event nodes corresponding to six verbs and their Maslow labels. We project the node embeddings, based on different encoders, to a 2-D space using t-SNE (Maaten and Hinton, 2008). We use shapes to represent verbs and colors to represent labels. In Fig. 3b and 3c, RoBERTa, pretrained on Whole-Word-Masking TAPT, was used. Nodes are word-contextualized, receiving the whole story (W-CTX-STORY) or the target sentence (W-CTX-SENT) as context. In these two cases, event nodes with the same verb (shape) tend to be closer. In Fig. 3a, we use ENG as the encoder to generate graph-contextualized embeddings (ENG-CTX). We observe that nodes with the same label (color) tend to be closer. In all cases, the embedding was trained using only the TAPT tasks, without task specific data. The ENG embedding is better at capturing entities' mental states, rather than verb information, as the graph structure is entity-driven.

Figure 4 makes this point quantitatively. We use 10-fold cross validation and report averaged results. The proximity between verbs and between labels are measured in two ways: cluster purity and KNN classification. For the cluster purity (Manning et al., 2008), we cluster the events using K-Means ($K = 5$), and calculate the averaged cluster

Models	Fulfilled			Unfulfilled			Average		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
ST-BOW	78.00	78.00	78.00	57.00	56.00	57.00	67.50	67.00	67.50
ST-ALL	78.00	79.00	79.00	58.00	56.00	57.00	68.0	67.50	68.00
ST-DISC	80.00	79.00	80.00	58.00	56.00	57.00	68.00	67.50	68.00
LR-BOW	69.00	65.00	67.00	53.00	57.00	55.00	61.00	61.00	61.00
LR-ALL	79.00	70.00	74.00	52.00	64.00	58.00	65.50	67.00	66.00
LR-DISC	75.00	84.00	80.00	60.00	45.00	52.00	67.50	64.50	66.00
BERT	81.75	75.90	78.72	57.95	66.23	61.82	69.85	71.06	70.27
BERT+ENG	81.99	83.06	**82.52**	65.33	63.64	**64.47**	73.66	73.35	**73.50**

Table 3: Results for the DesireDB task: identifying if a desire described in the document is fulfilled or not.



(a) ENG-CTX (b) W-CTX-STORY (c) W-CTX-SENT

Figure 3: t-SNE visualization of embeddings based on ENG and RoBERTa.

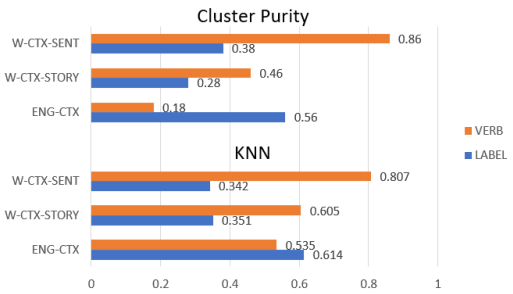


Figure 4: Cluster Purity and KNN Classification results for graph- and word-contextualized embeddings.

purity, defined as follows:

$$\frac{1}{N}\sum_{c\in C}\max_{d\in D}|c\cap d|, \tag{6}$$

where C is the set of clusters and D is either the set of labels or verbs.

For the graph contextualization, we can see that the labels have higher cluster purity than the verbs, while for the word contextualization, the verbs have higher cluster purity. This result aligns with our visualization. The KNN classification uses the learned embedding as a distance function. The KNN classifier performs better when classifying labels using the graph-contextualized embeddings, while it performs better using word-contexualized embeddings when classifying verbs. These results demonstrate that ENG can better capture the states of entities.

5 Conclusions

We propose an ENG model that captures implicit information about the states of narrative entities using multi-relational graph contextualization. We study three types of weakly-supervised TAPTs for ENG and their impact on the performance of downstream tasks, as well as symbolic inference capturing the interactions between predictions. Our empirical evaluation was done over two narrative analysis tasks. The results show that ENG can outperform other strong baselines, and the contribution of different types of TAPT is task-dependent. In the future, we want to connect different TAPT schemes and downstream tasks, and explore constrained representations.

6 Acknowledgements

We thank the reviewers for their efforts and insights. This work was partially funded by the NSF and DARPA ASED program under contracts CNS-1814105 and 13000686.

References

Muhammad Abdul-Mageed and Lyle Ungar. 2017. Emonet: Fine-grained emotion detection with gated recurrent neural networks. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 718–728.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction.

Kai-Wei Chang, Wen-tau Yih, Bishan Yang, and Christopher Meek. 2014. Typed tensor decomposition of knowledge bases for relation extraction. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1568–1579.

Snigdha Chaturvedi, Dan Goldwasser, and Hal Daumé III. 2016. Ask, and shall you receive? understanding desire fulfillment in natural language text. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2697–2703.

Muhao Chen, Hongming Zhang, Haoyu Wang, and Dan Roth. 2020. What are you trying to do? semantic typing of event processes. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 531–542.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. pages 4171–4186.

David Elson. 2012. Dramabank: Annotating agency in narrative discourse. In *LREC*, pages 2813–2819.

Radhika Gaonkar, Heeyoung Kwon, Mohaddeseh Bastan, Niranjan Balasubramanian, and Nathanael Chambers. 2020. Modeling label semantics for predicting emotional reactions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4687–4692, Online. Association for Computational Linguistics.

Amit Goyal, Ellen Riloff, and Hal Daumé III. 2010. Automatically producing plot unit representations for narrative text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 77–86.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. 2017. Tracking the world state with recurrent entity networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Clayton J Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth international AAAI conference on weblogs and social media*.

Yangfeng Ji, Chenhao Tan, Sebastian Martschat, Yejin Choi, and Noah A. Smith. 2017. Dynamic entity representations in neural language models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1830–1839, Copenhagen, Denmark. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

I-Ta Lee and Dan Goldwasser. 2019. Multi-relational script learning for discourse relations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4214–4226.

I-Ta Lee, Maria Leonor Pacheco, and Dan Goldwasser. 2020. Weakly-supervised modeling of contextualized event embedding fordiscourse relations. In *Findings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Wendy G Lehnert. 1981. Plot units and narrative summarization. *Cognitive science*, 5(4):293–331.

Zhongyang Li, Xiao Ding, and Ting Liu. 2018. Constructing narrative event evolutionary graph for script event prediction. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4201–4207. ijcai.org.

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.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Christopher D Manning, Hinrich Schütze, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*. Cambridge university press.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60.

A. H. Maslow. 1943. A theory of human motivation. *Psychological Review*, 50:370–396.

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.

Maria Leonor Pacheco and Dan Goldwasser. 2021. Modeling Content and Context with Deep Relational Learning. *Transactions of the Association for Computational Linguistics*, 9:100–119.

Debjit Paul and Anette Frank. 2019. Ranking and selecting multi-hop knowledge paths to better predict human needs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3671–3681, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Robert Plutchik. 1980. A general psychoevolutionary theory of emotion. *Theories of emotion*, 1:3–31.

Rashmi Prasad, Eleni Miltsakaki, Nikhil Dinesh, Alan Lee, Aravind Joshi, Livio Robaldo, and Bonnie L Webber. 2007. The penn discourse treebank 2.0 annotation manual.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Elahe Rahimtoroghi, Jiaqi Wu, Ruimin Wang, Pranav Anand, and Marilyn Walker. 2017. Modelling protagonist goals and desires in first-person narrative. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 360–369, Saarbrücken, Germany. Association for Computational Linguistics.

Hannah Rashkin, Antoine Bosselut, Maarten Sap, Kevin Knight, and Yejin Choi. 2018. Modeling naive psychology of characters in simple commonsense stories. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2289–2299, Melbourne, Australia. Association for Computational Linguistics.

Steven Reiss. 2004. Multifaceted nature of intrinsic motivation: The theory of 16 basic desires. *Review of General Psychology*, 8:179–193.

Maarten Sap, Ronan LeBras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2018. Atomic: An atlas of machine commonsense for if-then reasoning. *arXiv preprint arXiv:1811.00146*.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer.