Life is a Circus and We are the Clowns: Automatically Finding Analogies between Situations and Processes

Oren Sultan
The Hebrew University of Jerusalem
oren.sultan@mail.huji.ac.il

Dafna Shahaf
The Hebrew University of Jerusalem
dshahaf@cs.huji.ac.il

Abstract

Analogy-making gives rise to reasoning, abstraction, flexible categorization and counterfactual inference – abilities lacking in even the best AI systems today. Much research has suggested that analogies are key to non-brittle systems that can adapt to new domains. Despite their importance, analogies received little attention in the NLP community, with most research focusing on simple word analogies. Work that tackled more complex analogies relied heavily on manually constructed, hard-to-scale input representations. In this work, we explore a more realistic, challenging setup: our input is a pair of natural language *procedural texts*, describing a situation or a process (e.g., how the heart works/how a pump works). Our goal is to automatically extract entities and their relations from the text and find a mapping between the different domains based on *relational similarity* (e.g., blood is mapped to water).

We develop an interpretable, scalable algorithm and demonstrate that it identifies the correct mappings 87% of the time for *procedural texts* and 94% for *stories* from cognitive-psychology literature. We show it can extract analogies from a large dataset of procedural texts, achieving 79% precision (analogy prevalence in data: 3%). Lastly, we demonstrate that our algorithm is robust to paraphrasing the input texts[1].

1 Introduction

The ability to find parallels across diverse domains and transfer ideas across them is one of the pinnacles of human cognition. The *analogous reasoning* process allows us to abstract information, form flexible concepts and solve problems based on our previous experience (Minsky, 1988; Hofstadter and Sander, 2013; Holyoak, 1984) – abilities that current AI systems still lack. Many researchers have suggested that analogy-making is essential for engineering non-brittle AI that can robustly generalize and adapt to diverse domains (Mitchell, 2021b).

Surprisingly, despite analogy's important role in the way humans understand language, the problem of recognizing analogies has received relatively little attention in NLP. Most works have focused on SAT-type of analogies ("*a* to *b* is like *c* to *d*"), with recent works (Linzen, 2016; Ushio et al., 2021; Brown et al., 2020) showing that models still struggle with abstract, loosely defined relations.

In this work, we focus on a different type of analogies: analogies between *situations* or *processes*. Here the input is two domains (e.g., heart and pump) and the goal is to map objects from the base domain to objects from the target domain. Importantly, the mapping should rely on a common *relational structure* rather than object attributes, making it challenging for NLP methods.

The most influential work in this line of research is Structure Mapping Theory (SMT) (Gentner, 1983). In SMT and much of its follow-up work (Falkenhainer et al., 1989; Turney, 2008; Forbus et al., 2011), domain descriptions are sets of statements in a highly structured language, e.g.,

```
CAUSE(PULL(piston),CAUSE(GREATER(PRESSURE(water),
PRESSURE(pipe)),FLOW(water,pipe)))
```

Many have argued that too much human creativity is required to construct these inputs, and the analogy is already expressed in them (Chalmers et al., 1992). They are also brittle and hard-to-scale.

In our work, we explore a more realistic, challenging setup. Our input is a pair of two *procedural texts*, describing a situation or a process, expressed in natural language. We develop an algorithm to automatically extract entities and their relations from the text and find *a mapping* between the different domains based on *relational similarity*.

For example, the two texts in Figure 1 explain the animal cell to students through an analogy to a fac-

[1]Data and code are available in our GitHub repository https://github.com/orensul/analogies_mining

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 3547 - 3562
December 7-11, 2022

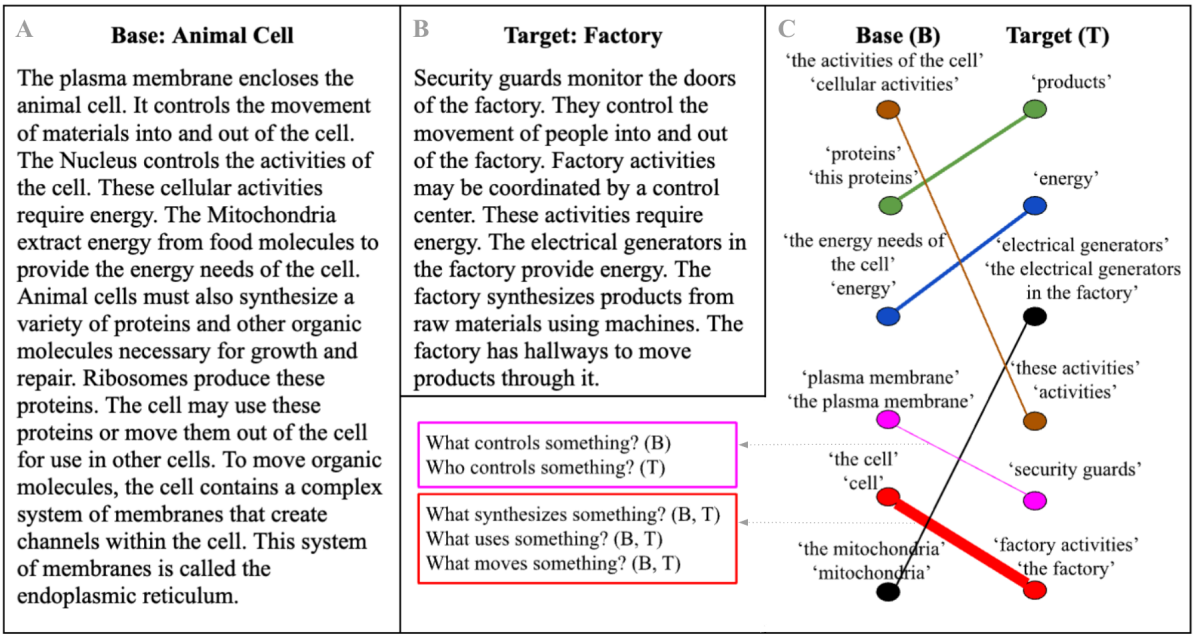


Figure 1: A+B: Example input – two analogous texts, describing the animal cell (base) and factory (target), adapted from Mikelskis-Seifert et al. (2008). C: Our algorithm's output. The nodes are entities (clusters of text spans; for the sake of presentation, we show up to two spans per cluster). Edge width represents similarity between entities in terms of the roles they play in the text. For example, the boxes on the left illustrate the similar roles associated with the red and the pink entities. All the mappings our algorithm found are correct, but two are missing (ribosomes/machines and endoplasmic reticulum/hallways). Showing the mapping along with its justification (the similar roles) renders our output *easy to interpret*.

tory (adapted from Mikelskis-Seifert et al. (2008))[2]. Note that in our setting entities and relations are not given upfront, but need to be extracted; they can appear multiple times, often expressed in different terms. Also, entities can play multiple roles throughout the text.

Figure 1C shows our algorithm's output. Nodes are entities (clusters of text spans – mostly correct, except for "factory activities"), and edge width represents similarity in terms of roles entities play. For example, the boxes on the left illustrate the similar roles associated with the red and the pink entities (represented as questions), showing that "plasma membrane" and "security guards" share one role (control something), and "cell" and "factory" share three. Importantly, including the justification as part of the output renders our algorithm *interpretable*. We note that all the mappings our algorithm found are correct, but two are missing (ribosomes/machines and endoplasmic reticulum/hallways). **Our contributions are:**

- We present a novel setting in computational analogy – mapping between *procedural texts* expressed in natural language. We develop a scalable, interpretable method to find mappings based on *relational similarity*.
- Our method identifies the correct mappings 87% of the time for *procedural texts* from ProPara dataset (Dalvi et al., 2018). Although not designed for them, our method achieves 94% precision on cognitive-psychology *stories* (Gentner et al., 1993; Ichien et al., 2020).
- We demonstrate our method can be used to mine analogies in the ProPara dataset, achieving 79% precision at the top of the ranking, when analogy prevalence in data is $\sim 3\%$.
- We show our method is robust to *paraphrasing* the input texts.
- We release data and code, including the found mappings at `https://github.com/orensul/analogies_mining`.

We hope this work will pave the way for further NLP work on computational analogy.

2 Problem Formulation

Our framework is based on Gentner's structure mapping theory (SMT) (Gentner, 1983). The central idea of SMT is that an analogy is a mapping of

[2]Note that this example is from a textbook, given for simplicity of presentation, and thus the texts are particularly well-aligned. Our method can handle much more variation.

objects from a base domain $\mathcal{B}$ into a target domain $\mathcal{T}$ that is based on a common *relational structure*, rather than object attributes. In our setup, the input is two *procedural texts* describing a situation or a process, expressed in natural language.

Entities. First, we need to extract entities from the texts. Let $\mathcal{B} = \{b_1, ..., b_n\}$ and $\mathcal{T} = \{t_1, ..., t_m\}$ be the entities in the base and the target, respectively. In our setting, entities are noun phrases. For example, in the animal cell (Figure 1), some entities include plasma membrane, animal cell, nucleus, energy, mitochondria, proteins, ribosomes.

Relations. Let $\mathcal{R}$ be a set of relations. A relation is a set of ordered entity pairs. In this work, we focus on verbs from the input text, but other formulations are possible. Intuitively, relations should capture that the mitochondria *provides* energy to the cell (in $\mathcal{B}$), just like electrical generators *provide* energy to the factory (in $\mathcal{T}$). Slightly abusing notation, let $\mathcal{R}(e_1, e_2) \subseteq 2^{\mathcal{R}}$ be the set of relations between e_1 and e_2. For example, $\mathcal{R}(cell, proteins)$ contains {synthesize, use, move}. Note that $\mathcal{R}$ is an asymmetric function, as the order matters.

Similarity. Let *sim* be a similarity metric between two sets of relations, $sim : 2^{\mathcal{R}} \times 2^{\mathcal{R}} \rightarrow [0, \infty)$. Intuitively, we want similarity to be high if the two sets *share many distinct* relations. For example, {provide, destroy}, should be more similar to {supply, ruin} than to {destroy, ruin} as the last set does not include any relation similar to provide. Given a pair of entities $b_i, b_j \in \mathcal{B}$ and a pair of entities $t_k, t_l \in \mathcal{T}$, we define a similarity function measuring how similar these pairs are, in terms of the relations between them. Since *sim* is asymmetric, we consider both possible orderings:

$$sim^*(b_i, b_j, t_k, t_l) = sim(\mathcal{R}(b_i, b_j), \mathcal{R}(t_k, t_l)) + sim(\mathcal{R}(b_j, b_i), \mathcal{R}(t_l, t_k)) \quad (1)$$

Objective. Our goal is to find a mapping function $\mathcal{M} : \mathcal{B} \rightarrow \mathcal{T} \cup \perp$ that maps entities from base to target. Mapping into $\perp$ means the entity was not mapped. The mapping should be *consistent* – no two base entities can be mapped to the same entity. We look for a mapping that maximizes the relational similarity between mapped pairs:

$$\mathcal{M}^* = \arg\max_{\mathcal{M}} \sum_{\substack{j \in [1, n-1] \\ i \in [j+1, n]}} sim^*(b_j, b_i, \mathcal{M}(b_j), \mathcal{M}(b_i))$$

If b_i or b_j maps to $\perp$, sim^* is defined to be 0.

3 Analogous Matching Algorithm

Our goal in this section is to find the best mapping between $\mathcal{B}$ and $\mathcal{T}$. Our algorithm consists of four phases: we begin with a basic **text processing** (Section 3.1). Then, we **extract potential entities and relations** (Section 3.2). Since entities can be referred to in multiple ways, we next **cluster** the entities (Section 3.3). Finally, we find a **mapping** between clusters from $\mathcal{B}$ and $\mathcal{T}$(Section 3.4).

We note that our goal in this paper is to present a new task and find a reasonable model for it; many other architectures and design choices are possible and could be explored in future work.

3.1 Text Processing

We begin by chunking the sentences in the input. As our next step is structure extraction, we first want to resolve pronouns. We apply a lightweight *co-reference* model (Kirstain et al., 2021) which generates clusters of strings (e.g, *the plasma membrane*, *plasma membrane*, *it*) and replace all pronouns by a representative from their cluster – the shortest string which is not a pronoun or a verb.

3.2 Structure Extraction

Analogy is based on relational similarity; thus, we now extract relations from the text. This naturally falls under Semantic Role Labeling (SRL) (Gildea and Jurafsky, 2002) – identifying the underlying relationship that words have with the main verb in a clause. In particular, we chose to use QA-SRL (FitzGerald et al., 2018). This model receives a sentence as input and outputs questions and answers about the sentence. See Table 1 for example questions and answers. Intuitively, the spans identified by QA-SRL as answers form the entities, and similar questions that appear in both domains (e.g., "what provides something?") indicate that the two entities (mitochondria, generators) may play similar roles.

We chose to use QA-SRL since it allows the questions themselves to define the set of roles, with no predefined frame or thematic role ontologies. Recent studies show that QA-SRL achieves 90% coverage of PropBank arguments, while capturing much implicit information that is often missed by traditional SRL schemes (Roit et al., 2020).

We focus on questions likely to capture useful relations for our task. We filter out "When" and "Why" questions, "Be" verbs, and questions and answers with a low probability (see Appendix A).

Text	Verb	Question	Answer
The animal cell	provide	what provides something?	the mitochondria
		what provides something?	mitochondria
		what provides something?	food molecules
		what does something provide?	the energy needs of the cell
The factory	provide	what provides something?	the electrical generators
		what provides something?	the electrical generators in the factory
		what does something provide?	energy

Table 1: Snippet from QA-SRL output on our cell/factory texts.

3.3 Clustering Entities

In classical computational analogy work, entities are explicitly given, each with a unique name ("cell"). However, in our input, entities are often referred to in different ways (*"the animal cell"*, *"the cell"*, *"cell"*), which might confuse the mapping algorithm. Therefore, in this step we merge those different phrasings, resulting in a new, more refined set of entities. Since we do not know in advance the number of clusters, we use Agglomerative Clustering (Zepeda-Mendoza and Resendis-Antonio, 2013). We manually fine-tuned the linkage threshold that determines the number of clusters (see Appendix B.2 for details). We denote the resulting clusters of entities as $\mathcal{B} = \{b_1, ..., b_n\}$ and $\mathcal{T} = \{t_1, ..., t_m\}$. Figure 2 shows the output clusters for the animal cell text. Most entities are identified correctly, although not all (e.g., clusters 5 and 7 should merge).

3.4 Find Mappings

Our problem definition (Equation 1) assumes we know all relations between the entities. However, our extraction algorithm is not perfect – e.g., it cannot detect relations described across sentences, or using complex references. Consider these sentences (slight paraphrases of the original texts):

"Animal cells must also produce proteins and other organic molecules necessary for growth and repair. Ribosomes are used for this process" / "The factory synthesizes products from raw materials using machines"

Ideally, we would like to infer that *ribosomes* produce *proteins* and *machines* synthesize *products*. QA-SRL only gives us partial information, but it is still useful. For example, both *proteins* and *products* are associated with similar questions (what is produced?, what is synthesized?), hinting that they might play similar roles. Thus, we propose a *heuristic approach* to approximate Equation 1.

Intuitively, the similarity score between two entities b_i, t_k is high if the similarity between their associated *questions* is high (for example, *cell* and *factory* have multiple distinct similar questions). We define this as the sum of cosine distances over their associated questions' SBERT (Reimers and Gurevych, 2019) embeddings. We filter out distances below a similarity threshold (see Appendix B.1). If there exists a sentence involving b_i and b_j and another sentence involving t_k and t_l, such that those sentences were used *both* for mapping b_i to t_k and b_j to t_l using *the same verb*, a complete *relation* was found. In this case, we increase the score of both mappings. There are multiple ways to do so; we found that a simple schema of adding a constant α to the similarity of (b_i, t_k) and (b_j, t_l) works well in practice (see Appendix B.2 for parameters).

We observe that questions are mostly of similar length (in ProPara, ∼1/3 of the questions have 3 words, ∼1/3 have 4 words, ∼1/6 have 2 words and ∼1/6 have 5 words). Note that the entities are not part of the questions.

Beam Search. After computing all similarities, we use beam search to find the mapping $\mathcal{M}^*$ (see Appendix B.2 for parameters).

Figure 1 shows our algorithm's top mapping for the factory/cell example. Edge weights represent similarity. All the mappings our algorithm found are correct, but two are missing (ribosomes/machines and endoplasmic reticulum/hallways). Being able to show the user the relations that led to the output mapping renders our method easy to interpret. We name it **Find Mappings by Questions (FMQ)**.

1) 'system of membranes', 'a complex system of membranes', 'this system of membranes', 'membranes', 'complex system of membranes'. **2)** 'food molecules', 'organic molecules'. **3)** 'energy', 'the energy needs of the cell'. **4)** 'these proteins', 'proteins'. **5)** 'animal cells', 'animal cell', 'the animal cell'. **6)** 'these cellular activities', 'cellular activities', 'the activities of the cell'. **7)** 'the cell', 'cell'. **8)** 'nucleus', 'the nucleus'. **9)** 'endoplasmic reticulum', 'the endoplasmic reticulum'. **10)** 'mitochondria', 'the mitochondria'. **11)** 'channels'. **12)** 'the plasma membrane', 'plasma membrane'. **13)** 'the activities'. **14)** 'ribosomes'. **15)** 'movement of materials', 'the movement of materials'.

Figure 2: The result of agglomerative clustering on animal cell text. Most of the entities were identified correctly, but not all (e.g., clusters 5 and 7 should merge).

4 Experiments

Our research questions are as follows:

- **RQ1**: Can we leverage our algorithm for *retrieving* analogies from a large dataset of procedural texts?
- **RQ2**: Does our algorithm produce the correct mapping solution?
- **RQ3**: Is our algorithm *robust* to paraphrasing the input texts?

We chose to test our ideas on the **ProPara** dataset (Dalvi et al., 2018) of crowdsourced paragraphs describing processes. Prompts (e.g., "What happens during photosynthesis?") were given to 1-6 workers each. We used the ProPara training set (390 paragraphs). See Appendix D.1 for two example paragraphs and our algorithm's mapping. Experiments run from a laptop. Runtime is few seconds for a pair of paragraphs on average.

4.1 Mining Analogies

One of the reasons for developing a metric for analogous similarity between paragraphs is to be able to *retrieve* analogies from a large corpus (**RQ1**).

To find analogies, we wish to *rank* all ∼76K possible pairs (over the 390 ProPara paragraphs), so that analogies rise to the top. We expect very few pairs of paragraphs to be truly analogous, while the number of pairs that might happen to have one or two strong entity matches could be significantly higher. Thus, we prefer a mapping involving more entities, even though their scores are not very strong. We chose a simple ranking formulation that balances between the number of mappings and their strength – multiplying the number of mappings by the median similarity, $|\mathcal{M}| \cdot median(\mathcal{M})$.

To the best of our knowledge, there is no baseline that solves our task. We first compare our method, **FMQ**, to **SBERT** (Reimers and Gurevych, 2019), a well-known method to derive semantically meaningful sentence embeddings for tasks like semantic textual similarity. The ranking is based on cosine similarity between paragraph embeddings.

The second baseline we use is a simpler variant of our method we call **Find Mappings by Verbs (FMV)**. FMV is identical to FMQ, but when finding a mapping (Section 3.4) we compute the similarity between the *verbs of the questions* instead of between the questions themselves. As verbs represent *relations*, which are a core part of analogies, this baseline is meant to test the additional benefit of using the questions extracted by QA-SRL.

We rank the 76K possible pairs via all three methods. We annotate the top 100 pairs, as well as 40 pairs from all quartiles (bottom, middle, 25% and 75%), resulting in a total of 260 annotated pairs from each list (702 unique). The main intersections are between FMQ and FMV (25% top, 95% bottom), and between FMQ and SBERT (11% top).

Labels. If the texts are not analogous to each other, we use the **Not analogy** label. Analogies are divided into **Self analogy** (entities and their roles are identical), **Close analogy** (a close topic, entities from a similar domain), **Far analogy** (unrelated topics with different entities), and **Sub-Analogy** (only a part of one process is analogous to a part of the other; should contain at least two similar relations.) See Table 2 for examples of different analogy types. Notice that we only show the prompt, but annotators could see the full paragraphs.

Annotation. We had an expert (member of our team) annotate the 702 *unique* pairs from the three lists in a double-blind fashion. As this is not an easy annotation, we performed two checks to assess the clarity and consistency of our annotation scheme.

First, another expert from our team (highly familiar with analogies) annotated a sample of the data (containing all labels), achieving 90% agreement, with Cohen's Kappa of 0.74 for the 2-labels and 0.88 for the 5-labels. Next, we recruited 15 volunteer annotators (graduate students in CS, most with a basic knowledge of analogies). We first trained the annotators, showing them two examples

Prompt in Base	Prompt in Target	Analogy Type
Describe how oxygen reaches cells	What do lungs do?	*Self analogy*
How does rain form?	How does snow form?	*Close analogy*
How does the digestive system work?	How does weathering cause rocks to break?	*Far analogy*
How does a solar panel work?	What happens during photosynthesis?	*Far analogy*
What happens during photosynthesis?	How does a virus infect an animal?	*Not analogy*

Table 2: Examples of different types of analogies, from the top of our method (FMQ) ranking on ProPara.

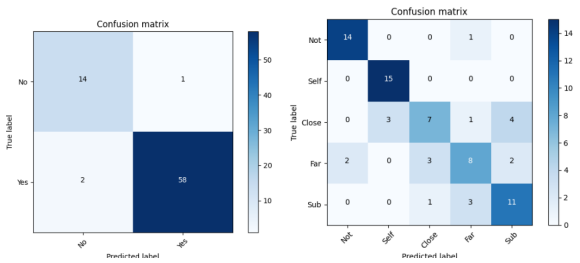


Figure 3: Confusion matrices comparing 15 annotators to an expert annotator. **Left:** 2-labels granularity (analogy/not-analogy). **Right:** 5-labels (No, Self, Close, Far, Sub). We see very high agreement on binary labels, and that *Self analogy* is the easiest for our annotators.

for each label, along with the correct label and an explanation. Annotators discussed the examples with the experimenter. We sampled from our expert's annotation 5 pairs for each label, resulting in 25 pairs of paragraphs. Each annotator received 5 pairs, s.t. each pair is assigned to 3 annotators.

When treating our expert's annotations as ground-truth, annotators' accuracy was 0.96 for the binary (analogy/non-analogy) task, and 0.73 for the 5-class task (27% scored perfectly, 33% had one mistake). Figure 3 shows the confusion matrices. We can see a very high agreement in the binary task, which is most important in our setting, as our goal is to mine analogies; we have created the fine-grained labels for conducting more in-depth analysis of the found analogies. Out of the analogy types, we see that self-analogy was the easiest for annotators. Annotators also mentioned it was sometimes hard to distinguish between *Far* and *Sub* analogies. Fleiss Kappa is 0.82 for the 2-labels and 0.58 for the 5-labels. We conclude from the two sanity checks that our annotation schema is overall effective, and the expert annotations can be used for evaluation.

Results. All methods had zero analogies in the 25%, middle, 75% and bottom samples; the only analogies found were at the top-100. At the top, SBERT found 100% analogies, FMQ reached 79% and FMV – 72%. At first glance, it seems like SBERT is winning. However, a closer look at analogy types (Table 3) shows it mostly finds *self-analogies* – paragraphs describing the same process (which makes sense, as it is designed for textual similarity), which are technically analogies, but not very interesting ones. On the other hand, FMQ and FMV did manage to find many far, close and sub-analogies, which is our goal.

Method	Not	Sub	Self	Close	Far
SBERT	0	0	89	11	0
FMV	28	15	26	20	11
FMQ	21	16	29	18	16

Table 3: Different type of analogies found in top-100 of the ranking. SBERT found mostly self-analogies, while FMQ and FMV found more interesting analogies.

Method	P	AP	NDCG
FMV (@25)	0.68	0.36	0.4
(@50)	0.72	0.37	0.41
(@75)	0.71	0.36	0.43
(@100)	0.72	0.36	0.43
FMQ (@25)	**0.96**	**0.5**	**0.57**
(@50)	**0.84**	**0.43**	**0.52**
(@75)	**0.77**	**0.39**	**0.47**
(@100)	**0.79**	**0.4**	**0.49**

Table 4: *Precision@k*, *AP@k* and *NDCG@k*. FMQ is consistently better than FMV in all the three metrics.

To estimate the prevalence of analogous pairs in the data, we randomly sampled 100 more pairs and had the expert annotate them, finding 3% analogies (one sub-analogy, one self analogy and one far analogy), confirming the hardness of the task.

Table 4 compares FMV and FMQ in terms of information retrieval metrics – *Precision (P)@k*, *Average Precision (AP)@k* and *Normalized Discounted Cumulative Gain (NDCG)@k*. For NDCG, we defined gains of 0, 1, 2, 3, 4 for not, sub, self, close and far respectively. We can see FMQ is consistently better than FMV in all metrics, supporting

our intuition that questions are more useful than verbs alone. See Appendix C.1 for full plots.

4.2 Evaluating the Mappings

In the previous section, we saw our method is able to identify analogies. However, we did not check that it indeed finds strong (true) mappings. In this section, we take a closer look at the mappings themselves and tackle **RQ2** – does our algorithm produce the correct mapping?

We chose 15 analogous pairs of paragraphs from ProPara, identified in the previous experiment (Section 4.1), equally divided between close, self and far (we did not take sub-analogies as they are harder for the annotators, sometimes with more than one correct answer). We assigned one paragraph to each of our 15 volunteer annotators and asked them to find the correct mapping between the entities.

We note that choosing pairs of paragraphs from the previous experiment might introduce some bias. To evaluate mappings, we need pairs of analogous texts. Randomly sampling and annotating ProPara pairs would be prohibitively expensive due to sparsity of analogies in the data; we do believe that analyzing a sample of the analogies found through our previous experiment, despite the potential bias, is still interesting and worth exploring.

In addition, we decided to try our algorithm on a different kind of data – **analogous stories** from cognitive-psychology literature (which does not suffer from the potential bias mentioned above). We used the Rattermann and Keane problems (Gentner et al., 1993; Ichien et al., 2020), resulting in 19 pairs of stories. We filtered out stories with dialogue or at most two mappings, leaving us with 14 pairs. See Appendix D.2 for an example pair, along with our algorithm's mapping. We assigned these stories to 14 annotators (one story per annotator), and asked them to do the same as above.

We instructed the annotators to find mappings between entities, and emphasized that the mappings should be consistent and based on the roles entities play in the texts. We showed them two examples of correct mappings with explanations. One user provided an invalid mapping for ProPara, and we discarded his annotation.

We consider the annotators' labels as ground truth, and the algorithm's mappings as predictions. We compare the performance of FMQ and FMV (Section 4.1). Again, to the best of our knowledge, there is no baseline for computing mappings.

Dataset	Method	P	R	F1
ProPara	FMV (@1)	0.48	0.33	0.39
	FMQ (@1)	**0.82**	**0.64**	**0.72**
	FMV (@3)	0.58	0.40	0.47
	FMQ (@3)	**0.87**	**0.67**	**0.76**
Stories	FMV (@1)	0.64	0.46	0.54
	FMQ (@1)	**0.88**	**0.68**	**0.77**
	FMV (@3)	0.73	0.52	0.61
	FMQ (@3)	**0.94**	**0.76**	**0.84**

Table 5: Evaluating the mappings of our method (FMQ) and FMV in terms of Precision (P), Recall (R) and F1 score. We compare the metrics on top-1 and top-3 solutions of beam search. FMQ outperforms FMV across all metrics.

Results. Table 5 shows precision, recall and F1 score for top-1 and top-3 solutions (from beam search). Our method (FMQ) achieves higher precision and recall than FMV on both datasets. It has a very high agreement with the annotators (P@3 of 0.87 in ProPara and 0.94 in stories). We believe this is the result of the richer information provided by the questions (e.g., the answers to "what is produced?" and "where is produced?" are probably not analogous, despite the same verb).

We note that the results on the stories are better for both methods, which is probably due to the fact that the stories are written as analogies in the first place, with explicit parallels between them.

Recall analysis. We analyzed FMQ's recall errors on stories and found that recurring sources of error include filtering "Where" questions, "How" questions and "Be" verbs. Refer to Appendix A for a short discussion of our filtering design choices. These error patterns apply to ProPara as well.

4.3 Robustness to Paraphrases

Our method heavily relies on the way the input texts are phrased. In this experiment, we examine robustness to *paraphrasing* (**RQ3**).

Automatic paraphrases. We focus on the ProPara dataset. We chose ten paragraphs which are not analogous to each other and generated four paraphrases using *wordtune*, a large-scale language model[3]: two expansions and two abridgments (examples in Appendix E.1). This results in 50 paragraphs, or 1225 pairs. We note that 80% of paraphrased sentence pairs contained different verbs.

We labeled the 100 pairs that came from the

[3] https://wordtune.com

same original paragraph as an analogy (in fact, they are *self-analogies*), and the rest as non-analogy. Then, we rank all pairs via SBERT, FMV and FMQ. Table 6 shows *precision@k* for top 100 (full plot in Appendix E.3). Both FMV and FMQ achieve very high precision (near-perfect for the first 50, and very high at 100), demonstrating that our method is relatively *robust* to *paraphrases*. As a reality check, SBERT achieves near-perfect results, which makes sense as these are self-analogies, and it is designed for textual similarity.

Responses to the same prompt. In the ProPara dataset, the same prompt is sometimes given to multiple authors. We now explore whether our algorithm can recognize those (self) analogies. Again, we take ten non-analogous paragraphs given to at least five authors, and randomly choose five authors for each, resulting in 1225 pairs of paragraphs, with 100 labeled as analogies. Unlike the previous experiment, the labels here are much *noisier*, as authors given the same prompt can focus on different aspects or granularity, resulting in non-analogous paragraphs (e.g., when describing the human life cycle, some authors mention zygotes and embryos, and others mention teenagers and adults. See Appendix E.2).

Table 6 shows *precision@k* results (full plot in Appendix E.3). Precision is reasonable, but lower than automatic paraphrases. The labels themselves are noisy, so it is harder to assess performance. Note that *recall@100* = *precision@100* in both automatic paraphrases and responses to the same prompt, as there are 100 true positives.

Error analysis. Looking at our model's false negatives, we see mostly pairs of paragraphs describing the same topic from different points of view, which is really a mistake in the ground truth (see example in Appendix E.4). We note that some annotators reported it was impossible to notice two paragraphs were self-analogies without seeing the (identical) prompts. SBERT is not much affected by this, as it looks at the entire text, while our methods are "blind" to the entities. Another interesting source of false negatives is mistakes introduced by wordtune (e.g., expanding "the water builds up" to "Nitrates build up in the body of the water"). For false positives, we identify several sources of error, such as non-analogous texts with similar verbs, QA-SRL handling of phrasal verbs ("take care", "take off"), repeating verbs, and extraction issues (for example, the sentence "Water, ice, and wind hit rocks" lead to singleton entities and "water, ice, and wind", resulting in double-counting).

Method	P@25	P@50	P@75	P@100
FMV	1.00	0.98	0.93	0.83
FMQ	1.00	0.98	0.89	0.74
SBERT	1.00	1.00	1.00	1.00
FMV	0.64	0.48	0.33	0.28
FMQ	0.64	0.5	0.44	0.38
SBERT	1.00	1.00	0.97	0.92

Table 6: *Precision@k* (P@k) in the *automatic paraphrasing* (top) and the *same prompt* task (bottom). Our methods achieve very high results in the *paraphrasing* task. The results in the *same prompt* task are moderately high, perhaps due to noisy ground truth (pairs about the same topic erroneously tagged as analogies). SBERT performs almost perfectly, as it sees the entire text, while our methods are blind to the entities.

5 Related Work

Computational analogy. Classical analogy-making approaches are typically categorized as symbolic, connectionist and hybrid. See (French, 2002; Mitchell, 2021a; Gentner and Forbus, 2011) for a comprehensive review.

In the NLP community, most work focused on simple word analogies. This area has gained popularity after showing that word embeddings can model some relational similarities in terms of word vector offsets (*"king - man + woman = queen"* (Mikolov et al., 2013)). Recent works (Linzen, 2016; Ushio et al., 2021; Brown et al., 2020) show that the offset method only works for some types of relations, struggling with complex, abstract ones.

In a different setting, LRME (Turney, 2008) took as input two sets of entities (base, target) and tried to extract a common relational structure between them, mining relations from a large web corpus. Unlike our work, their setting focused on commonsense relations (e.g., electrons revolve around the nucleus), and could not handle either procedural texts (where entities go through multiple stages) or relations that are situation-specific. Also, LRME requires entities and relations to be expressed exactly the same across domains, making it brittle.

Other works combined NLP and crowds to find analogies between products (Kittur et al., 2019; Hope et al., 2017). The models do not explicitly define entities and relations, but instead extract *schemas* and match based on them.

Aligning texts. Multi-text applications often need to model redundancies across texts. There has been much work in this area, exploiting graph-based representations (Cui et al., 2020; Yu et al., 2021), QA-SRL (Brook Weiss et al., 2021), and other semantic structures. This work is in similar spirit to ours, but the crucial difference is that in our task the algorithm is blind to entities, and only focuses on aligning the relations.

Procedural texts. Understanding procedural text has a long history in NLP, with a lot of work focusing on event extraction, tracking what happens to entities throughout the text (Henaff et al., 2017; Tandon et al., 2018; Bosselut et al., 2018). More recently, (Dalvi et al., 2019) extended these frameworks, to also understand *why* some actions need to happen before others. To the best of our knowledge, the task of finding analogous mapping between procedural texts is novel.

6 Conclusions and Future Work

Analogies can facilitate learning and problem-solving, helping people apply their prior experience to new situations. Much research has suggested that analogy-making is necessary for AI systems to robustly generalize and adapt to new contexts.

In this work we explored a complex, challenging analogy-finding setting: our input is a pair of natural language procedural texts, describing a situation or a process. We presented a novel, scalable and interpretable method to extract entities and relations from the text and find a mapping between entities across the domains. We show that our method successfully identifies the correct mappings between the domains for both procedural texts from ProPara, and stories from cognitive-psychology literature. We demonstrate our method can be used to mine analogies from ProPara, including far, non-trivial analogies. Lastly, we show that our method is robust to *paraphrases* the input texts.

In the future, we plan to improve our relation extraction and augment the text with commonsense knowledge to account for relations that do not appear explicitly in the text. We also plan to extend our algorithm to take the *order* of actions into account, and to apply our method to new domains, such as *legal texts* and *recipes*. Two particularly interesting applications are (1) *education*, where analogies can help a teacher explain a complex concept, and (2) *computer-assisted creativity*, where engineers and designers could find inspiration in distant domains.

We release our dataset, including the found analogies and their mappings (https://github.com/orensul/analogies_mining). We hope this work will encourage the development of novel NLP methods for computational analogy.

Acknowledgements We thank the anonymous reviewers for their constructive comments, the Hyadata Lab and NLP-HUJI members for their thoughtful remarks, and all the participants in our experiments for their efforts. This work was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant no. 852686, SIAM).

Limitations

- **Relation extraction**: as discussed in Section 3.4, QA-SRL misses some relations (e.g., those that are expressed across multiple lines). This reduces the effectiveness of our method. For this reason, we also expect our method to work best on more technical descriptions (where there are actions and entities that can be tracked), and less on paragraphs with a narrative style.
- **Insensitivity to order of actions**: our method does not take the order in which actions took place into consideration. For example, a sequence of actions and its reverse sequence would look analogical to the model.
- **Handling of phrasal verbs**: QA-SRL does not handle phrasal verbs well, reducing phrases such as 'take care" and "take off" to the verb "take" ("what takes something?").
- **Language**: Our datasets contain solely English texts. The results may differ in other languages.

Ethics Statement

To the best of our knowledge, the datasets used in our work do not contain sensitive information. The cognitive psychology dataset is somewhat antiquated, and might be considered to reinforce some stereotypes; we note we do not train any model on this data, but only use it for evaluation.

References

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. *ArXiv*, abs/1711.05313.

Daniela Brook Weiss, Paul Roit, Ayal Klein, Ori Ernst, and Ido Dagan. 2021. QA-align: Representing cross-text content overlap by aligning question-answer propositions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9879–9894, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

David J Chalmers, Robert M French, and Douglas R Hofstadter. 1992. High-level perception, representation, and analogy: A critique of artificial intelligence methodology. *Journal of Experimental & Theoretical Artificial Intelligence*, 4(3):185–211.

Peng Cui, Le Hu, and Yuanchao Liu. 2020. Inducing alignment structure with gated graph attention networks for sentence matching. *arXiv preprint arXiv:2010.07668*.

Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: A challenge dataset and models for process paragraph comprehension. *NAACL*.

Bhavana Dalvi, Niket Tandon, Antoine Bosselut, Wen tau Yih, and Peter Clark. 2019. Everything happens for a reason: Discovering the purpose of actions in procedural text. *ArXiv*, abs/1909.04745.

Brian Falkenhainer, Kenneth D Forbus, and Dedre Gentner. 1989. The structure-mapping engine: Algorithm and examples. *Artificial intelligence*, 41(1):1–63.

Nicholas FitzGerald, Julian Michael, Luheng He, and Luke Zettlemoyer. 2018. Large-scale qa-srl parsing. *arXiv preprint arXiv:1805.05377*.

Kenneth Forbus, Jeffrey Usher, Andrew Lovett, Kate Lockwood, and Jon Wetzel. 2011. Cogsketch: Sketch understanding for cognitive science research and for education. *Topics in Cognitive Science*, 3(4):648–666.

Robert M French. 2002. The computational modeling of analogy-making. *Trends in cognitive Sciences*, 6(5):200–205.

Dedre Gentner. 1983. Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7(2):155–170.

Dedre Gentner and Kenneth D Forbus. 2011. Computational models of analogy. *Wiley interdisciplinary reviews: cognitive science*, 2(3):266–276.

Dedre Gentner, Mary Jo Rattermann, and Kenneth D Forbus. 1993. The roles of similarity in transfer: Separating retrievability from inferential soundness. *Cognitive psychology*, 25(4):524–575.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic Labeling of Semantic Roles. *Computational Linguistics*, 28(3):245–288.

Mikael Henaff, Jason Weston, Arthur D. Szlam, Antoine Bordes, and Yann LeCun. 2017. Tracking the world state with recurrent entity networks. *ArXiv*, abs/1612.03969.

Douglas R Hofstadter and Emmanuel Sander. 2013. *Surfaces and essences: Analogy as the fuel and fire of thinking*. Basic books.

Keith J Holyoak. 1984. Analogical thinking and human intelligence. *Advances in the psychology of human intelligence*, 2:199–230.

Tom Hope, Joel Chan, Aniket Kittur, and Dafna Shahaf. 2017. Accelerating innovation through analogy mining. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 235–243.

Nicholas Ichien, Hongjing Lu, and Keith J. Holyoak. 2020. Verbal analogy problem sets: An inventory of testing materials. *Behavior Research Methods*.

Yuval Kirstain, Ori Ram, and Omer Levy. 2021. Coreference resolution without span representations. In *ACL/IJCNLP*.

Aniket Kittur, Lixiu Yu, Tom Hope, Joel Chan, Hila Lifshitz-Assaf, Karni Gilon, Felicia Ng, Robert E Kraut, and Dafna Shahaf. 2019. Scaling up analogical innovation with crowds and ai. *Proceedings of the National Academy of Sciences*, 116(6):1870–1877.

Tal Linzen. 2016. Issues in evaluating semantic spaces using word analogies. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pages 13–18, Berlin, Germany. Association for Computational Linguistics.

Silke Mikelskis-Seifert, Ute Ringelband, and Maja Brückmann. 2008. *Four decades of research in science education: from curriculum development to quality improvement*. Waxmann Münster, Germany.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Marvin Minsky. 1988. *Society of mind*. Simon and Schuster.

Melanie Mitchell. 2021a. Abstraction and analogy-making in artificial intelligence. *Annals of the New York Academy of Sciences*, 1505(1):79–101.

Melanie Mitchell. 2021b. Abstraction and analogy-making in artificial intelligence. *Annals of the New York Academy of Sciences*, 1505.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Paul Roit, Ayal Klein, Daniela Stepanov, Jonathan Mamou, Julian Michael, Gabriel Stanovsky, Luke Zettlemoyer, and Ido Dagan. 2020. Controlled crowdsourcing for high-quality qa-srl annotation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7008–7013.

Niket Tandon, Bhavana Dalvi, Joel Grus, Wen tau Yih, Antoine Bosselut, and Peter Clark. 2018. Reasoning about actions and state changes by injecting commonsense knowledge. In *EMNLP*.

Peter D Turney. 2008. The latent relation mapping engine: Algorithm and experiments. *Journal of Artificial Intelligence Research*, 33:615–655.

Asahi Ushio, Luis Espinosa-Anke, Steven Schockaert, and Jose Camacho-Collados. 2021. Bert is to nlp what alexnet is to cv: can pre-trained language models identify analogies? *arXiv preprint arXiv:2105.04949*.

Xueli Yu, Weizhi Xu, Zeyu Cui, Shu Wu, and Liang Wang. 2021. Graph-based hierarchical relevance matching signals for ad-hoc retrieval. In *Proceedings of the Web Conference 2021*, pages 778–787.

Marie Lisandra Zepeda-Mendoza and Osbaldo Resendis-Antonio. 2013. *Hierarchical Agglomerative Clustering*, pages 886–887. Springer New York, New York, NY.

A QA-SRL Ignore QA criteria

Here we explain our criteria for ignoring QA which we parsed from the QA-SRL model.

- When the probability of the question (according to QA-SRL) is less or equal to our question probability threshold (see B.2 for the chosen threshold).
- When the probability of the answer (according to QA-SRL) is less or equal to the answer probability threshold (see B.2 for the chosen threshold). It means that the probability this is the correct span is too low.
- When the question is not one of "what", "who" and "which". The rationale is that we focus on questions most likely to capture useful relations for our task.
- When the question's verb is "be". The rationale is that "be" is not indicative enough. For example, if we have "X was something" on one text and "Y was something" on the other text, it does not indicate that X and Y play similar roles.
- When the answer contains a verb or does not contain a noun. In this case, it does not an entity according to our definition.
- When the answer is a pronoun.

B Parameters Fine-tuning

Here we discuss how we choose the value for different parameters which are used in the experiments. We note our goal in this paper was to come up with a proof-of-concept system, and further parameter tuning might improve the results.

B.1 Cosine Similarity Threshold

To determine the best cutoff for cosine similarity threshold between questions (FMQ) or between verbs (FMV), we did the following: We sampled 15 pairs of verbs (from ProPara paragraphs) in every range of threshold (intervals of 0.05 from 0 to 1 for *questions* and for *verbs*), then manually labeled the pairs of verbs as similar or not, and chose the threshold of **0.5**. The threshold was chosen to balance between precision (percentage of correct samples from all samples passing the threshold) and an estimation of recall (percentage of correct samples from all correct samples that do manage to pass the threshold), computed using samples from all intervals. For the similarity of the questions we did the same process, but instead of verbs, we sampled questions. We found that **0.7** is the best cutoff. See Figure 4 for different thresholds and their similarity accuracy.

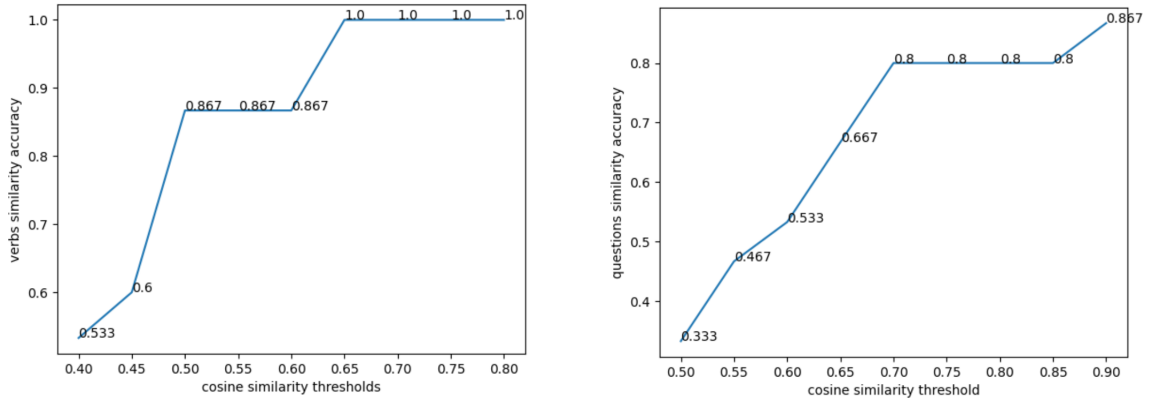


Figure 4: *Verbs* similarity accuracy (on the left) and *questions* similarity accuracy (on the right) for different cosine similarity thresholds. We can see that a threshold of 0.5 for the *verbs* and 0.7 for the *questions* achieves accuracy of at least 0.8.

B.2 Other Parameters

We choose the values for the following parameters by *manual fine-tuning*. For the QA-SRL settings, we set the *answer probability threshold* to **0.05** (range checked: 0.0-0.15) and *question probability threshold* to **0.1** (range checked: 0.0-0.15, intervals of 0.05), optimizing for F1-score. We set the Agglomerative clustering *distance* parameter to be **1.0** (range checked: 0-2 with a step size of 0.5), optimizing for cluster purity compared to a ground-truth clustering on several examples. For the SBERT embedder model, we used the pre-trained model *msmacro-distilbert-base-v4* (Reimers and Gurevych, 2019). When a complete *relation* was found, we add α of **1.0** (range checked: 0-2 with a step size of 1) for both pairs of entities in base and target, again optimizing on several left-out examples. We set the beam search size to **7** (range checked: 1-10).

C Analogies Mining

C.1 FMQ / FMV top@k comparison

Here we compare our method (FMQ) and FMV on the top of their lists according to information retrieval metrics. See Figure 5 for three plots which compare between FMV and FMQ methods on the different metrics.

D Mappings Evaluation Examples

D.1 Pair of ProPara paragraphs

See Figure 6 for two example paragraphs from ProPara and our algorithm's mapping.

D.2 Pair of analogous stories

See Figure 7 for the texts and our algorithm output on the general/surgeon analogous stories.

E Algorithm Robustness

Here we show examples of automatic paraphrases made by wordtune as well response of different authors to the same prompt. We show the results of our algorithm's robustness. Finally, we analyze the main errors sources.

E.1 Automatic Pharaphases

We use the tool *wordtune*, which was developed by AI21 for our *paraphrases'* evaluation on paragraphs from ProPara, as part of our robustness evaluation (4.3). *Wordtune* has several features, some of them are *wordtune short* for rewriting the text in a shorter way, and *wordtune expand* for rewriting the text with longer sentences. See Figure 8 for an example.

E.2 Responses to the Same Prompt

In addition to paraphrases, we also want to challenge our system with different text versions of different authors on the same topic. This is a part of the robustness evaluation (4.3). To do so, we take four different versions for one original paragraph. See Figure 9 for an example.

E.3 Robustness Precision

See Figure 10 for two plots of *precision@k* for k between 1-100 of the top-ranked pairs of paragraphs for *paraphrases* and same *prompt* evaluations.

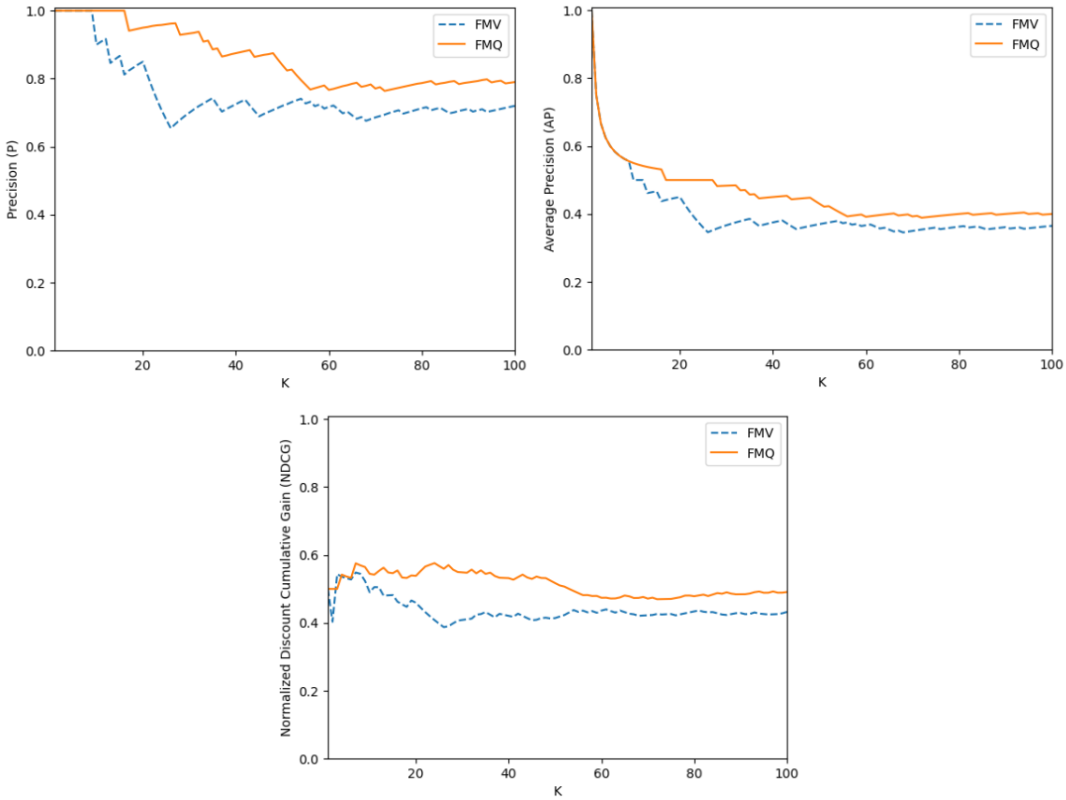


Figure 5: Comparison between FMQ and FMV methods in terms of *Precision (P) @k*, *Average Precision (AP) @k* and *Normalized Discounted Cumulative Gain (NDCG) @k* (k from 1 to 100). As we can see, FMQ is consistently better than FMV in all the three metrics.

E.4 Same Prompt, No Similar Verbs

See Figure 11 for two paragraphs with the same prompt, but with different verbs.

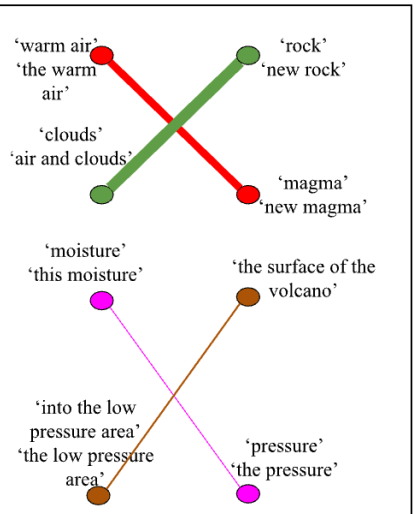
PROMPT: Describe the process by which hurricanes form Warm water floats up from the ocean. A pocket of low pressure air is created near the surface of the ocean. Warm air from outside areas pushes into the low pressure area. The warm air keeps rising and forms clouds. The wind is getting fed moisture that evaporates from the ocean. This moisture causes the swirling air and clouds to grow bigger. As it gets bigger the wind rotates faster.	**PROMPT: What causes a volcano to erupt?** Magma rises from deep in the earth. The magma goes into volcanos. The volcanos pressure the magma upwards. The pressure causes the magma to push through the surface of the volcano. The lava cools. The lava forms new rock. New magma is pressured to the surface of the volcano. The volcano bursts through the rock that formed after the last eruption.	

Figure 6: Left and Middle: two *procedural texts* paragraphs from ProPara, describing hurricanes (base) and volcano eruptions (target). Right: Our algorithm's output. The nodes are entities (clusters of text spans; for the sake of presentation, we show up to two spans). Edge width represents similarity between entities in terms of the roles they play in the text. According to the annotation, all mappings are correct.

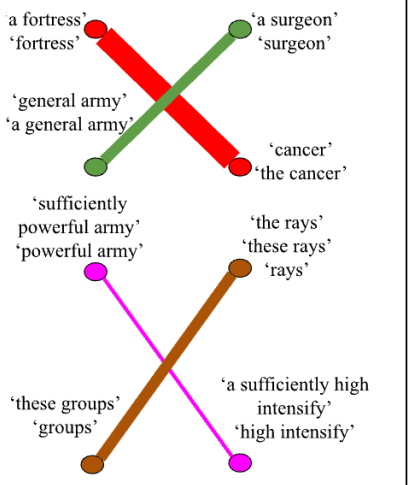
Base: The general A general was trying to destroy a fortress which was situated at the center of a country with roads leading to it, by using his army. He needed to use his army as a complete group in order to destroy the fortress. However, he could not march his army down a road to the fortress because the roads were mined to explode when large groups of men passed over them. After considerable thought, he knew just what to do. He divided his army up into small groups of men, and by sending these groups, simultaneously, from a number of different directions, they converged on the fortress, making up a sufficiently powerful army to destroy it.	**Target: The surgeon** A surgeon was trying to destroy a cancer which was situated in the central region of a patient's brain, by using a type of ray. He needed to use these rays at a high intensity in order to destroy the cancerous tissue. However, at such an intensity the healthy brain tissue will also be destroyed. After considerable thought, he knew just what to do. He divided the rays up into batches of low-intensity rays, and then by sending them, simultaneously, from a number of different directions, they converged on the cancer, making up a sufficiently high intensity to destroy it.	

Figure 7: Left and Middle: two analogous *stories* from cognitive psychology literature, describing a general trying to attack a fortress (base) and the surgeon trying to destroy a tumor (target). Right: Our algorithm's output. The nodes are entities (clusters of text spans; for the sake of presentation, we show up to two spans). Edge width represents similarity between entities in terms of the roles they play in the text. According to the annotation, the *precision* is perfect, but the *recall* is not as the following mappings are missing: "center of a country" to "central region of a patient's brain" and "roads" to "healthy tissue" (which we miss is because we filter "where" questions).

Original paragraph: How do lungs work?
You breathe air in. Air enters bronchial tubes. Air is then split into the bronchioles of each lung. Bronchioles have alveoli which are tiny air sacs. Alveoli is surrounded by many small blood vessels. Oxygen passes through alveoli into blood vessels. Blood leaves the lungs as it travels through the body. Blood carries carbon dioxide back to the lungs. Carbon dioxide released when you exhale.

Wordtune expand:
When you breathe in, you are taking in air. Through your bronchial tubes, air enters your lungs. After the air has passed through the bronchial tubes, it is divided into the bronchioles of each lung. Alveoli, which are tiny sacs of air, are situated in the bronchioles. The alveoli are surrounded by a big number of small blood vessels. It is through these blood vessels that oxygen moves into the alveoli. In the course of its journey through the body, the blood enters through the lungs. When blood returns to the lungs, it takes carbon dioxide along with it. It is this carbon dioxide that is released when you breathe out.

Wordtune short:
Breathing air in. Bronchial tubes obtain air. Lungs split air into bronchioles. Alveoli are tiny air sacs in the bronchioles. Small vessels nearby alveoli. Alveoli grab oxygen to blood vessels. As blood passes through the body, it leaves the lungs. CO2 is carried by blood to the lungs. CO2 is discharged when you breathe out.

Figure 8: An example of *paraphrases* – one long (top) and one short (bottom) made by *wordtune* for a paragraph describing how lungs work.

Original paragraph: Describe the life cycle of a human
A human baby develops in the womb of the mother. After 9 months in the womb the baby is born. It is an infant. The infant is dependent on its parents for everything. It drinks mother's milk for nourishment. From 3-8 years old the child is in early childhood. Adolescence is from roughly 9-18 years old. During adolescence the child is growing rapidly and maturing sexually. At 18 years, the child becomes an adult. Adults can reproduce and have babies.

V1:
A human is born. The human is an infant. The infant grows into a toddler. The toddler grows into a child. The child grows into a teenager. The teenager grows into an adult. The adult grows old. The human dies.

V2:
A human is born. The human is a child and learns. The human child grows into an adult. The adult uses its skills to survive. The human starts a new family and propogates. The human grows old. The human dies.

V3:
A zygote is formed via sexual reproduction. This zygtoe grows in the womb to become a fetus. After a typical 9-month period, a human is born. The human is an infant at this stage. The infant becomes a toddler, and learns to walk and speak. The toddler becomes a child. The child becomes a teenager after undergoing puberty. The teenager grows into an adult. The adult hits a peak, and development stops. Old age and eventually death occur.

V4:
A sperm fertilizes an egg. The egg forms into a fetus. 9 months passes as the fetus grows into an infant. The infant is born. The baby begins to grow into an adolescent. The adolescent turns into a young adult. The young adult learns and grows into a fully mature adult.

Figure 9: An example of different ProPara paragraphs with the same *prompt* (written by different authors) describing the human life cycle. We can see authors focus on different aspects or granularity (e.g., some authors mention zygotes and embryos, and others mention teenagers and adults).

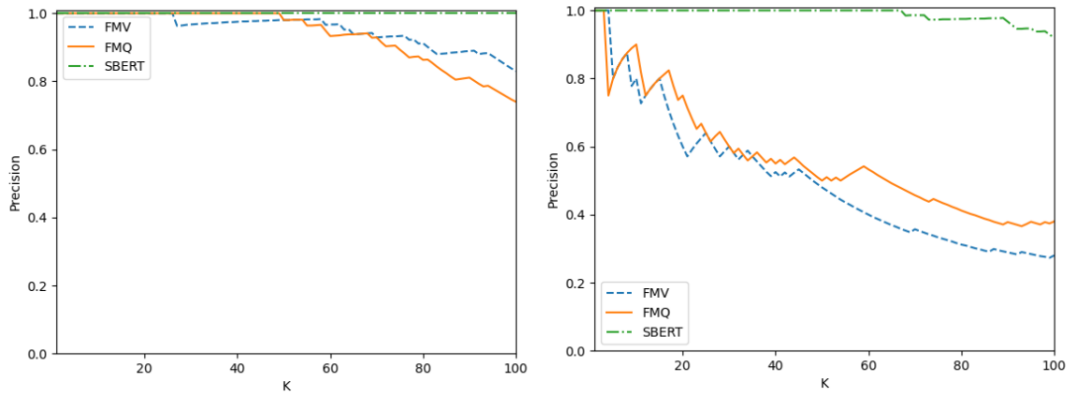


Figure 10: Left: *Precision@k* in the *automatic paraphrasing* task. Right: *Precision@k* in the *same prompt* task. Our methods achieve very high results in the *paraphrasing* task. The results in the *same prompt* task are moderately high, perhaps due to the noisier setting (many pairs tagged as analogies in our ground truth are about the same topic but not actually analogous). SBERT was able to perform almost perfectly, as it looks at the entire text, while our methods are blind to the entities.

V1: how does internal combustion engine work?
Air and fuel are **used** in the internal combustion engine. In an enclosed chamber, a mixture of air and fuel is **injected**. The mixture **ignites** and **turns** a piston that **pumps** up and down. This piston is **connected** to a crankshaft which **rotates** to **provide** the power. The burned gas is **pushed** out of the chamber.
V2: how does internal combustion engine work?
The piston **moves** down. Gasoline and air **go** into the engine. The piston **moves** back up. The gasoline and air are **compressed**. The spark plug **emits** a spark. The gasoline **explodes**. The explosion **forces** the piston down. The exhaust valve **opens**. Exhaust **goes** to the tailpipe.

Figure 11: An example of pair of paragraphs with the same prompt (written by different authors) which is ranked in the top 100 by SBERT (score of 0.54), but getting a zero score by our method, as these texts describe the same process ("How does internal combustion engine work") from a different point of view and with no *similar verbs* (verbs are in bold).