Some Notes on Modeling the Semantic Flow of Narratives

Daniel Khashabi
University of Illinois,
Urbana-Champaign
Urbana, IL, 61801, USA
khashab2@illinois.edu

Christopher J.C. Burges
Erin Renshaw **Andrzej Pastusiak**
One Microsoft Way
Redmond, WA, 98052, USA
{cburges,erinren,andrzejp}
@microsoft.com

Abstract

This report summarizes our investigations into modeling semantics and coherence in narratives, which has applications in many NLP tasks. We study MCTest, a recently designed task for machine comprehension, which contains multiple-answer questions about paragraph-long fictional stories. Our focus is on creating useful meaning-representations for modeling stories and interaction of characters in stories, which we explain thoroughly. The work on some final ideas is ongoing.

1 Introduction

Machine comprehension has long been one of the main goals in the Natural Language Processing (NLP) community, although there does not seem to be a consensus on the right way to measure the quality of the *comprehension*. Despite this, many believe that evaluation through properly designed tasks is the right way to make progress in inducing semantic information in computers (Burges, 2013).

Recently Richardson et al. (2013) introduced a challenge dataset of narratives, with multiple-choice questions. For human testers, the dataset is relatively easy; an elementary-school student is able to answer all of the questions with high accuracy, while the best known computer systems have performance of 69% [1], which shows a big gap between human performance and computer performance. As such, there is a significant headroom for improvement in this task.

An example from MCTest-160, is the following:

Story: "Sally had a very exciting summer vacation. She went to summer camp for the first time. She made friends with a girl named Tina. They shared a bunk bed in their cabin. Sally's favorite activity was walking in the woods because she enjoyed nature. Tina liked arts and crafts. Together, they made some art using leaves they found in the woods. Even after she fell in the water, Sally still enjoyed canoeing. She was sad when the camp was over, but promised to keep in touch with her new friend.

Sally went to the beach with her family in the summer as well. She loves the beach. Sally collected shells and mailed some to her friend, Tina, so she could make some arts and crafts with them. Sally liked fishing with her brothers, cooking on the grill with her dad, and swimming in the ocean with her mother.

The summer was fun, but Sally was very excited to go back to school. She missed her friends and teachers. She was excited to tell them about her summer vacation."

Question: Who went to the beach with Sally?
1) her sisters
2) Tina
3) her brothers, mother, and father
4) herself

As usual, the problem with naive supervised models is that, there is not enough labeled data available to learn everything. For this reason, we have tried to avoid early use of Machine Learning models in our investigation. Instead the focus is mostly on finding expressive meaning-representations which can be useful for modeling semantics, and interactions between animate entities in stories.

In addition to MCTest, there are other similar challenge tasks which need attentions from researchers in the community. An example task is coreference resolution on Winograd schema examples

[1] In this work, all of the numbers we report are on the MCTest-160.

(Levesque et al., 2011), which contains a carefully curated set of co-reference problem instances. Many of these instances need careful use of *world knowledge* and *causal* connection between events. There are other relevant works and tasks that we will refer to, in future sections. We believe there is a strong connection between these tasks, and the problem we are studying. The progress in one of them will highly benefit the other.

2 Relevant works

There is a plethora of work in Information Retrieval and Data Mining for visualization and extraction of meaningful patterns from massive data. For example, extracting *metro maps* from newswire or books (Shahaf et al., 2013; Shahaf and Horvitz, 2010; Shahaf and Guestrin, 2010). One of things we do is extraction of timeline of events for each animate entity in a relatively short story. Although very similar, but extraction of the map of interactions in small stories is (usually) much more brittle and is very different in nature from the aforementioned works.

In (Andreas and Klein, 2014), the model learns to ground language to path. The model is supervised with labeled data. The goal is very close to ours, which is grounding onto a useful representation, but which representation is most useful, is dependent of the input data, target task and scalability of the representation.

One other goal we have is finding relevant sentences to a given question (which with high probability contain the answer to it). This is relevant to the problem of *coherence* (Barzilay and Lee, 2004), as the question being asked and the target sentence need to topically cohere.

Many works try to model and learn text coherence directly. These systems are usually evaluated based on their ability to put together a shuffled story. (Foltz et al., 1998) use Latent Semantic Analysis to score semantic relatedness between sentences. An HMM-based model is used in (Barzilay and Lee, 2004). In (Barzilay and Lapata, 2008), an entity-based model is used to assess local textual coherence. In (Raghavan et al., 2014) the ordering of the events is directly tackled. They extract events, create timelines and align them by a cross narrative co-reference. More recently, (Li and Hovy, 2014) used neural networks for learning coherence.

In many scenarios understanding coherence needs perceiving facts beyond its surface text, e.g. knowledge implicitly mentioned by common-sense. There are many knowledge bases (e.g. `Freebase`) that contain many facts (e.g *Bill Gates was CEO of Microsoft.*). One of the major efforts to model causality and common sense is the `ConceptNet` (Speer and Havasi, 2012). [2] Usually the facts inside `conceptNet` are not mentioned anywhere (e.g. newswire), but we humans know them, since we have experienced them (common sense knowledge). For example, the fact that, *eating lunch* is (usually) *motivated by being hungry*, might be very hard to find in Wikipedia or newswire. But we know it, since we experience it everyday, and `ConceptNet` contains it. [3] Although `ConceptNet` (and similar resources) are be very valuable resources, the problem with using such knowledge-bases is that, they are very crude, sparse not easy to use practice. There needs to be a lot of engineering on top of the input facts so that one can get enough generalization for a desired problem.

There are series of works (Chambers and Jurafsky, 2008; Chambers and Jurafsky, 2009) that try to extract connection between verb predicates using unsupervised co-occurrence information. Similarly (Balasubramanian et al., 2013) extracts co-occurrence information at triple level. As a part of our final work, we propose a way to use triple co-occurrence information for clustering a story into coherent chunks, using triple-level information.

3 Some early thoughts on MCTest

Richardson et al. (2013) introduce a relatively simple baseline system, which is a combination of a sliding window, token-matching baseline and a metric baseline, where the metric is story-dependent. The sliding window baseline works by combining *question* + $option_i$ on the *story* and scoring the $option_i$ based on the number of the word matching the option. The $option_i$ with the highest score will be chosen as the answer. Although very simple, this baseline has been observed to be very powerful and

[2] See `http://conceptnet5.media.mit.edu/`.

[3] See `http://conceptnet5.media.mit.edu/web/c/en/hungry`

Question contains	Total number	Correctly answered by the baseline	Ratio
What type	3	1	.33
What time	1	1	1
What color	11	7.75	0.704
What	132	105.75	0.80
Where	19	14	0.73
When	5	4.25	0.85
Which	12	5	0.41
How many	8	3.75	0.46
How old	1	1	1
How	10	6.25	0.62
Who	40	25.75	0.64
Why	24	15.25	0.63
Can	2	1	0.5
None of the above	4	3	0.75

Table 1: The distribution of the type of questions, with the corresponding baseline performance.

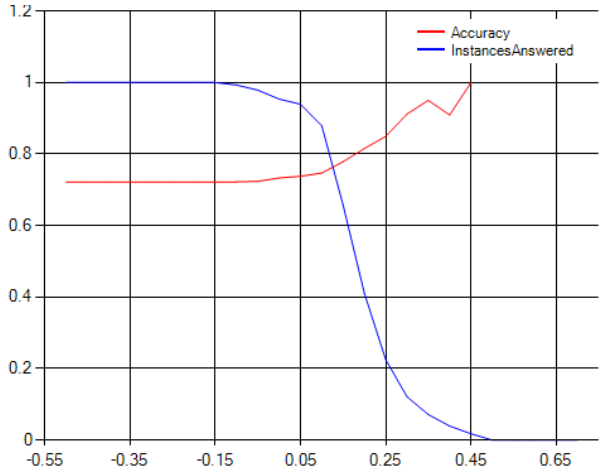
Question contains	Total number	Correctly answered by the baseline	Ratio
First	258	185.75	0.719
Last	1	1	1
n't / not	13	8	0.61

Table 2: The distribution multiple modifiers, with the corresponding baseline performance.

hard to improve by a significant margin (for details, see Algorithm *SW+D* in (Richardson et al., 2013)).

One possible way to make progress in designing a better Question Answering (QA) system for this challenge is a careful analysis of the type of questions in the data (Li and Roth, 2002). Table 1 and Table 2 show the distribution of the questions in the MCTest. The performance of the baseline on almost all of the question types, is very close to average overall performance. Given the spread of the errors, the idea of focusing on improving answers for few classes of question types doesn't seem to be very promising.

If the baseline system is working reasonably we would expect it to make more mistakes for questions that it has less confidence. In other words if the design is sensible, it is expected to do better in answering easier questions, than harder questions. Now given this baseline system, can we somehow separate the set of easy questions and hard questions for it? If we do so, we might be able to *abstain* from answering hard questions and pass them to stronger systems that are carefully designed for more complicated questions.



(a)

Figure 1: The *coverage* and *accuracy* vs. v_{th}. The scores, normalized with the length of the question.

The baseline system outputs a score for each option, which we assume is monotonic in the likelihood that the answer is correct. Consider the experiment in which, given a threshold v_{th}, we abstain from answering all of the questions for which the best option has a score less than v_{th}. Clearly the *coverage* (ratio of the questions answered by the baseline, and not abstained) decreases, as the value of the threshold v_{th} increases. In Figure 1 the blue curve shows the *coverage* of the questions being answered by the baseline. The red curve shows the accuracy of the system, on the subset of the questions not abstained. It can be seen that there is a monotonic increase in the accuracy of the system, as the coverage decreases. [4]

To summarize our observations from this experiment:

- Scores that we get from the baseline make sense, since smaller scores correspond to harder questions (on which we are more likely to make mistakes).

- Increased performance comes at the cost of losing substantial coverage. Almost 87% accuracy

[4]Except the very end of the red curve, which actually is not a good estimate of the accuracy due to the negligible number of the instances being answered by the baseline.

Figure 2: Stacking two systems; one for filtering sentences in story, and a second system to perform QA on the filtered story.

at 20% coverage (abstaining from around 80% of the questions).

Can we create a stronger system that could be stacked with the baseline system? In this report we do not have a good answer for this question, although our observations might be a good motivation for researchers to come up with such systems.

4 Filtering stories might help QA

We start this section with the question: *can filtering stories help QA*? In other words, given a relatively long story and a question, is it possible to choose a subset of story sentences, that contain the answer to the question (with high probability)?

Before designing any method for filtering stories we investigate the headroom for any possible improvement. Suppose we stack two systems as shown in Figure 2. The first system filters the sentences in the story, and the second system performs QA on the filtered stories. For now, we assume using the baseline in (Richardson et al., 2013) for the *System2* in Figure 2. Next we explain how we came up with an oracle for *System1*.

Suppose we manually remove irrelevant sentences from the stories. Given sets of *(story, question, options)* answered wrongly by the baseline, consider the following two schemes for filtering the stories:

- Scheme 1: Keep the sentences needed to answer the question, disregarding anything in the set of options.
- Scheme 2: Keep the sentences needed to answer the question, plus the sentences that have mentions of all the options (the misleading sentences).

After manually filtering the stories based on Scheme 1 and Scheme 2, we run the baseline system on them, and get the results in Table 3. The

Accuracy on baseline mistakes		
No pruning	Scheme 1	Scheme 2
0.0 %	38.77 %	14.13 %

Table 3: Running baseline on the instances for which the baseline makes mistakes, using different strategies for pruning the stories. The first column is included to stress the fact that pruning is done on the instances where the baseline makes mistakes.

pruning is done on the instances where the baseline makes mistakes (zero accuracy on this set). Keeping only the sentences needed for answering the question (Scheme 1), the baseline gets 38.77%. Keeping the misleading sentences in addition (Scheme 2) would decrease it by almost half. The reason for such drop is the existence of some words in the wrong options in the selected snippet which could mislead the baseline as it partly works based on sliding window of words matching. Although we use the baseline system as our QA engine, proper filtering of the sentences can give significant improvement. For this reason next we will propose multiple ways for filtering stories given its associated question.

Suppose we have an already trained-system, which given a question q and a sentence s, gives the score $score(s, q)$, which shows how likely it is that the sentence s contains the answer to the question q. Given this score we want to eliminate some of the sentences that do not contain the answer (with a high probability). For that purpose, we select the system in (Yih et al., 2013), which is one of the state-of-the-art QA systems.[5] This system scores a pair of *(Sentence, Question)*. Given any sentence $s_i \in$ *Story*, and question q, run the scoring system on (s_i, q) and get the score $score(s, q)$. For a given threshold v_{th}, keep all the sentences with the $score(s_i, q) \geq v_{th}$. When the value of the threshold is small enough (call it v_{small}), no sentence is eliminated, and the performance must be the same as that of the baseline. If the threshold v_{th} is big enough (call it v_{big}), no sentences are left and the performance is expected to be the lowest. The hope

[5] See: http://aclweb.org/aclwiki/index.php?title=Question_Answering_(State_of_the_art).

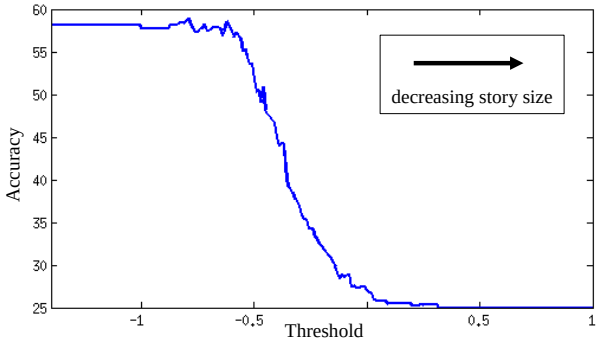


Figure 3: The performance of the baseline on the pruned stories. There is a very small performance improvement in average threshold values.

is to get considerable performance increase for some valued of threshold between v_{small} and v_{big}. The Figure 3 shows the performance of the baseline on the filtered stories, versus the value of the threshold v_{th}. There is a small jump in the accuracy, as we slide the threshold between v_{small} and v_{big}, but not considerable and not statistically significant.

The main reason to blame is the mismatch between the training data of (Yih et al., 2013) and MCTest. In addition, for many questions, the answer is spread across multiple sentences, which requires more careful treatment.

Although here we did not gain much from this experiment, we are a little hesitant to conclude that this idea is not worth further study. One potential flaw in our study is the mismatch between the datasets on which the scoring system is trained and tested on. Another issue is the careful normalization of the scores (which might be a function of input sentence length, question length, popularity of the tokens, etc).

5 Timeline based on co-reference resolution

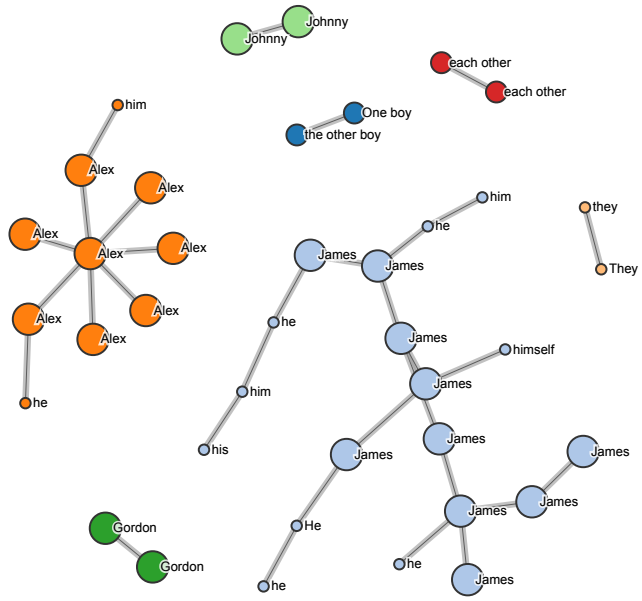
In this section, we turn to another way of representing the flow of knowledge in stories data, which might be useful for representing the information in the stories and in question answering. More specifically we will try to represent the knowledge in a narrative in the form of a timeline of the main characters in the story. Suppose we are given co-reference chains for a given story. Can we create a coherent set of event chains, for each animate entity?

An output of co-reference chains by the Illinois co-reference resolution system (Chang et al., 2013) is shown in Figure 4. As can be seen, some of the chains are about single entities (call it *singular-chain*), and some about a group of entities (call it *plural-chain*). For example, the chains $C_1 = \{Alex, him, \ldots\}$ (the orange chain) or $C_2 = \{James, he, him, \ldots\}$ (the light blue chain) are singular-chains. On the other hand, $C_3 = \{eachother, eachother\}$ or $C_4 = \{they, they\}$ chains are plural, since they refer to multiple people (more specifically C_3 and C_4 create the intersections between C_1 and C_2).

Since we want to model the timeline for single entities, we design a rule-based algorithm for connecting any plural-chain, to its relevant singular-chain. After connecting the chains, we generate the timeline for the story, using the singular-chains. Each singular-chain, has its own timeline, with events ordered as they appear in the story. The role of the plural-chains that are connected to multiple singular-chains, is to create intersection between events of the story. In other words, when two mentions of two different singular-chains belong to a common plural-chain, there is an intersection between two timelines in the sentences which contains the two mentions. If there is a plural chain which is not connected to any singular-chain, we treat it as singular-chain, and generate its own timeline.

Two example timelines generated by the above method are shown in Figure 5. There is a timeline associated with each animate entity (for example *Tina*, *Sally*, etc). During the course of the story, each animate entity goes through a set of events, which are denoted by white circles. When there is a an event shared between two (or more) entities, there is an intersection between their timelines. Our goal of generating the timelines for connecting events and interaction of animate entities is to create an informative representation for finding relevant sentences, given a question.

As can be seen in Figure 5, there are imperfections in the outputs of the timeline generation. Some timelines do not correspond to animate entities. There are a few redundant events in each timeline (and some events are missing). The reasons for the noise in the output are the followings:



Story: "Two best friends lived in a small town. They grew up together and did everything together. One boy was named James, and he was a cute, fat kid who was always picked on. And then the other boy was named Alex, and he was handsome and cool. They were opposites, but they didn't care. Alex would always protect James from the bullies because James was weak.

One day, James was being teased by the bullies, Gordon and Johnny, but Alex wasn't there to save him. James didn't know what to do and was scared. But then he remembered all of the times Alex stood up for him. James looked up to Alex and he thought of him as his hero. James wanted to protect himself and not have Alex always come to help. So this time, James got up and was brave. He stood up to Gordon and Johnny and told them that he wasn't scared. The bullies didn't know what to do and then gave up. They went away to look for more fun things to do. Alex saw the whole thing and was proud of James for being brave.

Now James and Alex protect each other and stand up for each other. They still are the best of friends."

Figure 4: co-reference chains for a sample story are visualized. Using a rule-based system we can use these chains to generate a timeline of events and interaction as shown in Figure 5.

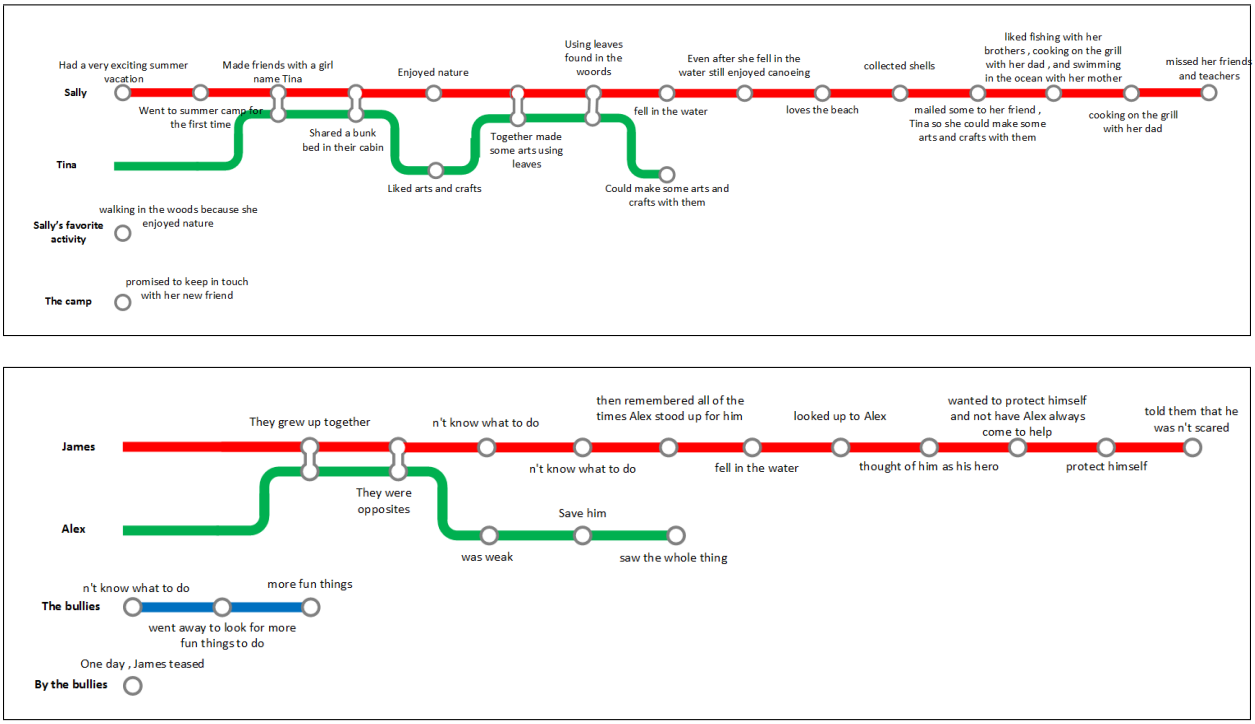


Figure 5: Two sample timelines generated using co-reference resolution. There is a timeline associated with each animate entity (for example *Tina*, *Sally*, etc). During the course of the story, each animate entity goes through a set of events, which are denoted by white circles. When there is a an event shared between two (or more) entities, there is an intersection between their timelines. This is created automatically using Illinois co-reference system and hand-built rules.

1. Mention detection: not detecting some mentions of entities, or detecting irrelevant words as mentions (wrong boundaries).

2. Co-reference resolution: improper clustering of the mentions.

3. The algorithm to connect co-reference chains: mistakes when connecting plural- and singular-chains.

5.1 Using timelines to help QA

As we observed in the previous section, timelines of story events are an expressive way of representing events and interactions between animate entities. But can we use this timeline to improve QA for a given story?

Suppose a tuple of (*Story; Question; Options*) are given. Given the *Story*, we extract the timeline for it. Suppose we receive the following question:

Question: What did Sally do this summer?

For many types of questions, it suffices to find the entity mention in the question, connect it to its corresponding timeline, and keep the sentences relevant to the selected timeline, which is another strategy to filter stories, as we mentioned before in Section 4.

Such selection of the sentences based on the timeline seems to be a reasonable filtering strategy for sentences, followed by our baseline question-answering system. The filtering step, reduces the size of the stories by 16.8%. If the filtering step is perfect (i.e. we do not remove any relevant sentences), we expect reasonable performance improvement. In our experiments, not much of improvement, however we reduced the size of the texts without losing much in performance. This is important when using a very powerful, but slow, question-answering system. Although in practice, there is not big difference in the accuracy of the baseline on the original stories, compared to the filtering strategy introduced here, this does not mean that filtering does not have any effect on the decisions. In fact, by removing some of the misleading sentences, the baseline corrects some of its mistakes (compared to the case without filtering). But instead, since the timeline is noisy, sometimes we mistakenly remove important sentences essential to answer the question, adding mistakes during the question-answering later on.

6 Event co-occurrence for timeline generation

Our experiments in the previous section for generating a timeline of events, were highly dependent on using co-reference resolution system. Here we summarize our efforts to remove this dependence on co-reference, by using co-occurrence information.

Suppose we change our definition of timeline slightly, and refine it: *"A proper chain in a timeline, contains a set of events logically following each other. Timeline chains might intersect with each other, via joint events, at some point in the story. Each character might have multiple timeline chains, depending on the coherency of the actions the character involved in."*.

To understand our motivation for this change of definition, consider the following ordered list of events:

(1:Jack wakes up) (2:Jack goes to bathroom) (3:Jack washes his face) (4:Jack opens his laptop) (5:Jack starts writing his final report) (6:Jack checks his Twitter)

Although all of the events have the same subject and all make sense with the current order, as humans and using our world-knowledge, we can understand a clustering of events (1,2,3) and events (4,5,6). Following this idea, we try to create timeline chains, based on their coherence, rather than merely having the same subject.

In (Balasubramanian et al., 2013) the authors have gathered co-occurrence data for tuples of the following form:

$$(Arg1; Relation; Arg2)$$

An example of such tuple is the following:

$$(\text{Abramoff; was indicted in; Florida})$$

Since there might be a lot of variations in the way the tuples might appear, a normalization of the arguments (such as entities to PERSON, etc) and the relation (e.g. stemming of the verb) is performed to

induce sparsity in the data. As an example, the above tuple can be normalized in the following form:

$$(\text{person; be indicted in; state})$$

After extraction of tuples and normalization, the co-occurrence information are gathered. Here is the overall algorithm summarized:

- Extract tuples using OLLIE [6].
- Stem the verb in the relation phrase
- Add additional tuples, by adding "types to Arg1 and Arg2. For example, given (he, cite, study), create 3 more instances:
  - (he, cite, ⟨activity⟩)
  - (⟨person⟩, cite, study)
  - (⟨person⟩, cite, ⟨activity⟩)
- Count the co-occurrences, for close enough events.

The authors run extraction of the tuples on New York Times section of Gigaword (1.8 M documents), gathering about 11M pairs of event co-occurrences. [7]

We can use the co-occurrence information between the normalized tuples. Using the co-occurrence information we can create the probability of transition between events:

$$p(e_j|e_i), \forall e_i \in \mathcal{E}_1, \forall e_j \in \mathcal{E}_2$$

such that:

$$\sum_{\forall e_j \in \mathcal{E}_2} p(e_j|e_i) = 1, \forall e_i \in \mathcal{E}_1$$

Given a story, extract all of the events in the story $\mathcal{E}_{story}$. For any pair of events in the story, extract event transition probability $p(e_2|e_1)$. The indicator variable $x(e_1 \rightarrow e_2)$ shows the directed connection from e_1 to e_2, if it is 1. We can formulate the generation of timelines as clustering edges of a graph with directed probability transitions, which can be formulated with ILP formulation (inspired by (Chang et al., 2011)).

$$\begin{cases} \max_x \sum_{e_i \in \mathcal{E}_1, e_j \in \mathcal{E}_2} x(e_1 \rightarrow e_2)\,(p(e_2|e_1) - \theta) \\ x(e_1 \rightarrow e_3) \geq x(e_1 \rightarrow e_2) + x(e_2 \rightarrow e_3) - 1, \\ \qquad\qquad\qquad\qquad\qquad\qquad \forall e_1, e_2, e_3 \\ x(e_1 \rightarrow e_2) \in \{0, 1\} \\ \theta \in (0, 1) \end{cases}$$

θ is a handle to control number of clusters.

Here we do not report any results on the last section, since the results of the experiments are not yet finalized and work on this is ongoing. There are a few obstacles for the idea that we refer to here. The biggest problem with this approach for clustering text into coherent subsets, is the lack of enough coverage of the Relgram dataset, over the triples extracted from the story. Another problem has roots in the limitation of representing knowledge in triples and estimating their *logical coherence* based on its co-occurrence information (or using frequency information to estimate *logicality*), which might not be good enough to give a reasonable set of transition probabilities between events; not everything that gets repeated more frequently is necessarily logical (or equivalently, not everything that does not get repeated, is illogical).

7 Future work and extensions

The major focus of the ongoing work is thorough experimental evaluation on generating of coherent chunks of stories using triple level information. A further step is to combine the result with event timeline generation, which is highly dependent on co-reference resolution. There many side issues here; like how good/bad of a representation is using triples, how to deal with the sparsity issue in the triple co-occurrence data, how to induce commonsense logic into the inference (like knowing that *something gets wet when it falls into water*), etc.

Extending timeline representation of narratives to spatial proximity (in which intersections correspond to spatial closeness; objects/entities can be, for example, close or far), can be a very useful representation of information in the stories. It turns out that this representation is actually very hard to extract.

[6] Available here: `http://knowitall.github.io/ollie/`.

[7] Available here: `http://relgrams.cs.washington.edu/`.

Similarly one can come up with different extensions of the timeline representations for narratives with different properties, other than spatial proximity or interaction.

Acknowledgments

We wish to thank Matt Richardson, Chris Meek, David Grangier, Scott Yih, Mrinmaya Sachan and Eric Horn for discussions and helpful comments. We also thanks Niranjan Balasubramanian for providing the code for extracting the Relgram tuples.

References

Jacob Andreas and Dan Klein. 2014. Grounding language with points and paths in continuous spaces.

Niranjan Balasubramanian, Stephen Soderland, Oren Etzioni Mausam, and Oren Etzioni. 2013. Generating coherent event schemas at scale. In *EMNLP*, pages 1721–1731.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 113–120, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

C.J.C. Burges. 2013. Towards the machine comprehension of text: An essay. Technical Report MSR-TR-2013-125, Microsoft Research.

Nathanael Chambers and Daniel Jurafsky. 2008. Unsupervised learning of narrative event chains. In *ACL*, pages 789–797. Citeseer.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 602–610, Stroudsburg, PA, USA. Association for Computational Linguistics.

K.-W. Chang, R. Samdani, A. Rozovskaya, N. Rizzolo, M. Sammons, and D. Roth. 2011. Inference protocols for coreference resolution. In *CoNLL Shared Task*, pages 40–44, Portland, Oregon, USA. Association for Computational Linguistics.

K.-W. Chang, R. Samdani, and D. Roth. 2013. A constrained latent variable model for coreference resolution. In *EMNLP*.

Peter W Foltz, Walter Kintsch, and Thomas K Landauer. 1998. The measurement of textual coherence with latent semantic analysis. *Discourse processes*, 25(2-3):285–307.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*.

Jiwei Li and Eduard Hovy. 2014. A model of coherence based on distributed sentence representation.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Preethi Raghavan, Eric Fosler-Lussier, Noémie Elhadad, and Albert M Lai. 2014. Cross-narrative temporal ordering of medical events.

Matthew Richardson, Christopher JC Burges, and Erin Renshaw. 2013. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *EMNLP*, pages 193–203.

Dafna Shahaf and Carlos Guestrin. 2010. Connecting the dots between news articles. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–632. ACM.

Dafna Shahaf and Eric Horvitz. 2010. Generalized task markets for human and machine computation. In *AAAI*.

Dafna Shahaf, Jaewon Yang, Caroline Suen, Jeff Jacobs, Heidi Wang, and Jure Leskovec. 2013. Information cartography: creating zoomable, large-scale maps of information. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1097–1105. ACM.

Robert Speer and Catherine Havasi. 2012. Representing general relational knowledge in conceptnet 5. In *LREC*, pages 3679–3686.

Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. 2013. Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1744–1753, Sofia, Bulgaria, August. Association for Computational Linguistics.