Extraction of Message Sequence Charts from Narrative History Text

by

Girish Palshikar, Sachin Pawar, Sangameshwar Patil, Swapnil Hingmire, Nitin Ramrakhiyan, Harsimran Bedi, Pushpak Bhattacharyya, Vasudeva Varma

in

Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT 2019)

Report No: IIIT/TR/2019/-1

Centre for Language Technologies Research Centre
International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2019

Extraction of Message Sequence Charts from Narrative History Text

Girish K. Palshikar **Sachin Pawar** **Sangameshwar Patil**
Swapnil Hingmire **Nitin Ramrakhiyani** **Harsimran Bedi**
{gk.palshikar,sachin7.p,sangameshwar.patil}@tcs.com
{swapnil.hingmire,nitin.ramrakhiyani,bedi.harsimran}@tcs.com
TRDDC, TCS Research and Innovation, India

Pushpak Bhattacharyya
pb@cse.iitb.ac.in
IIT Patna, India

Vasudeva Varma
vv@iiit.ac.in
IIIT Hyderabad, India

Abstract

In this paper, we advocate the use of Message Sequence Chart (MSC) as a knowledge representation to capture and visualize multi-actor interactions and their temporal ordering. We propose algorithms to automatically extract an MSC from a history narrative. For a given narrative, we first identify verbs which indicate interactions and then use dependency parsing and Semantic Role Labelling based approaches to identify senders (initiating actors) and receivers (other actors involved) for these interaction verbs. As a final step in MSC extraction, we employ a state-of-the art algorithm to temporally re-order these interactions. Our evaluation on multiple publicly available narratives shows improvements over four baselines.

1 Introduction

Narrative texts, particularly in history, contain rich knowledge about actors and interactions among them along with their temporal and spatial details. For such texts, it is often useful to extract and visualize these interactions through a set of inter-related timelines, one for each actor, where the timeline of an actor specifies the temporal order of interactions in which that actor has participated. *Message Sequence Chart (MSC)* is an intuitive visual notation with rigorous mathematical semantics that can help to precisely represent and analyze (Alur et al., 1996) such scenarios. Feijs (2000), and Li (2000) propose techniques to convert software requirements to MSC. Event timeline construction is a related task about inferring the temporal ordering among events, but where events are not necessarily interactions among actors (Do et al., 2012). Another related line of research is storyline or plot generation from narrative texts such as news stories or fiction (Chambers and Jurafsky, 2009; Vossen et al., 2015, 2016; Goyal et al., 2010; Kim et al., 2018), which uses different narratological output representations (not MSC), such as event sequences or story curves.

In this paper, we extract actors and their interactions from the given input history narrative text, and map them to actors and messages in the basic MSC notation. We generalize the previous work along several dimensions, and propose an *unsupervised* approach enriched with linguistic knowledge. MSC extracted from the given history text can be analyzed for consistency, similarity, causality and used for applications such as question-answering. For example, from the example in Table 1 we extract the MSC as shown in Figure 1, which can be used to answer questions like `"Whom did Napoleon defend the National Convention from?"`. To the best of our knowledge, this is the first work that uses MSC to represent knowledge about actors and their interactions in narrative history text. Our approach is general, and can represent interactions among actors in any narrative text (e.g., news, fiction and screenplays). We propose unsupervised approaches using dependency parsing and Semantic Role Labelling for extracting interactions and corresponding senders/receivers. We use a state-of-the-art tense based technique (Laparra et al., 2015) to temporally order the interactions to create the MSC.

2 Problem Definition

The input is a document D containing narrative text, and the desired output is an MSC depicting the interactions among the actors. No information about the actors or interactions is given as input; they need to be identified. For history narratives, we define an *actor* as an entity of type Person, Organization (ORG) or Location (LOC), which actively participates in various interactions

Proceedings of the First Workshop on Narrative Understanding, pages 28–36
Minneapolis, Minnesota, June 7, 2019.

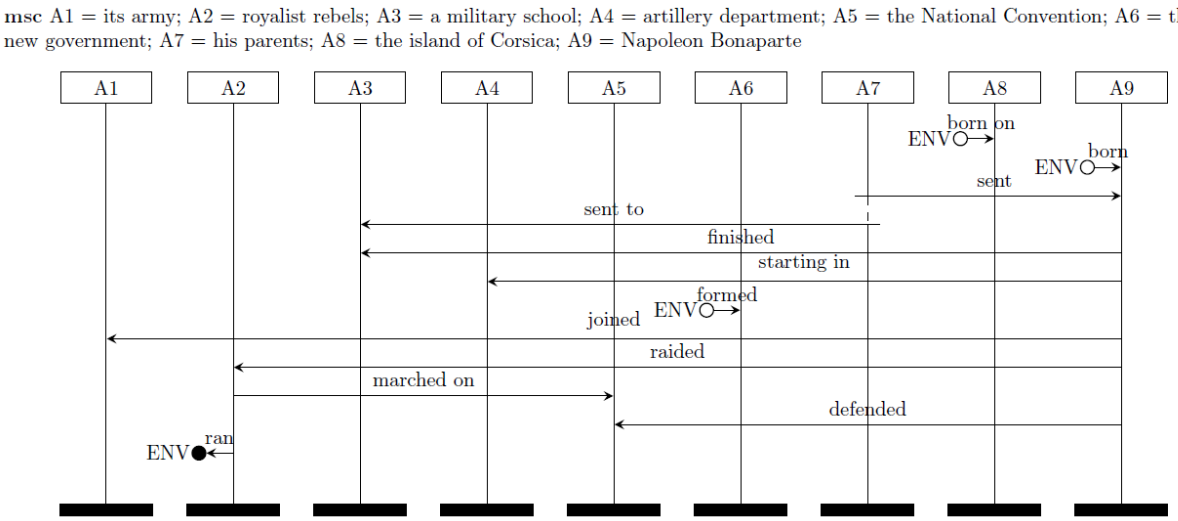

Figure 1: MSC for the example history text.

1. Napoleon Bonaparte was born in 1769 on the island of Corsica.
2. When he was 9 years old, his parents sent him to a military school.
3. He finished school in 1785 before starting in the artillery department.
4. When the new government was formed , Napoleon joined its army.
5. When royalist rebels marched on the National Convention in October 1795,
the young officer defended it.
6. The rebels then ran away in panic.
7. Three months earlier, Napoleon had raided the rebels.

Table 1: Sample narrative text. Implicit and explicit temporal expressions are underlined.

with other actors. The reason for including LOC entities as actors is that locations are important in history, and a timeline of events at a particular location provides an interesting perspective. Further, we also need to identify all coreferences of an actor and use a *canonical (i.e., a standardized, normalized*) mention for each; e.g., In Table 1, the actor mentions, `Napoleon Bonaparte`, `Napoleon`, `he` and `the young officer` refer to the actor `Napoleon Bonaparte`.

An *interaction* among actors is either (i) any *deliberate* (*intentional*) *physical action*, which is typically initiated by one or more actors and the remaining actors involved in it are affected by it in some way (e.g. `attacked`, `joined`), or (ii) *communication*, which results in passing of information or control among them (e.g. `announced`, `talked`).

We focus on interactions involving one or two actors. An interaction with itself involves only one actor; e.g., `the attackers fled`. When more than two actors are involved in an interaction (e.g., `Napoleon's parents sent him to a military school.`), we break it into several pairwise interactions, if possible. On the other hand, if the sender or receiver in an interaction are missing, we use a dummy actor *environment* (denoted by ENV) as the corresponding sender or receiver. For instance, in the sentence `"The rebels ran away in panic"`, there is no explicit receiver. So, as shown in Fig. 1, we use ENV as the receiver for the message, i.e., we create the message `(The rebels; ran; ENV)`.

Since we represent an interaction as a message in an MSC, the direction of the interaction is important. We assume the direction to be from the initiator of the interaction to the actor affected by it. However, some interactions can be directionless; e.g., `met, married`. In such cases, we show the subject of the sentence as sender of the message in MSC. Though our notion of an interaction is similar to an event, a key difference between them is that there is explicit and intentional involvement of actors in an interaction; e.g., `an earthquake` is an event but it is not an interaction.

Temporal ordering of messages is the important and culminating step in the overall process of automated MSC extraction from narrative text. We need to exploit temporal clues available in the input narrative text to derive the temporal order among the messages in the MSC.

2.1 Scope

In this paper, we focus on interactions expressed using verbs because most of the action events in a language are expressed using verbs. We consider interactions expressed using nouns as part of future work. Not all interactions in history narratives are important for creating an MSC. E.g., *mental actions*, such as `felt cheated`, `came to know`, `assumed`, `considered`, `envisioned`, are not considered as interactions. Copula verbs and verbs denoting a state of an object or actor also do not trigger an interaction and hence, such verbs are ignored.

3 MSC Extraction

3.1 Actor Identification

We first make one pass through the text and identify all the actors who are involved in one or more interactions. We group all co-referring mentions of an actor into a set, and choose one *canonical mention* as a representative on the MSC. One complication can occur due to *complex actors*, which is an actor that contains multiple actors, one of which is *independent* and the others are *dependent* and serve to elaborate on the independent actor; e.g., `his parents, military school, the army of the new government`. We need to identify a complex actor as a whole, and not its constituent actors separately. We use the algorithm in (Patil et al., 2018) to identify an actor and all its coreferents.

3.2 Interaction Identification

Typically the input text mentions many different interactions, and identifying each verbal interaction is required, omitting non-interactions as discussed in Section 2.1. A simple algorithm classifies each verb in the given sentence as an *action verb* or a *communication verb* (and ignores other types of verbs) using WordNet hypernyms of the verb itself or its nominal forms. For example, for the verb `defended`, one of its nominal forms, `defence`, has the category **act** in its hypernym tree; so it is classified as an action verb.

Since we are focusing on interactions that have already occurred, we focus on verbs in the past tense. In some cases, a verb not in the past tense, should also be considered as having past tense; e.g., in `Growing up in rural Hunan, Mao described his father as a stern disciplinarian`, "`Growing`" should be considered to be in the past tense. To achieve this, we systematically *propagate* the past tense to other verbs using linguistic rules. To detect verbs in past tense, we traverse the dependency parse tree of the input sentence in breadth-first-search (BFS) manner. A verb having POS tag of *VBD* is definitely in the past tense. A verb with *VBG* or *VBN* POS tag is considered to be in past tense if: (i) it is a child of another verb tagged with *VBD*; or (ii) it is the parent of an auxiliary verb tagged with *VBD*. An infinitive verb is deemed to be in past tense if it has a governor in the dependency tree with dependency relation either **advcl:to** or **xcomp** and the governor is tagged with *VBD*. In the above sentence, `described` is tagged with *VBD* and hence it is in past tense; `Growing` is tagged with *VBG* and is child of `described` in the dependency parse tree; hence, it is also considered to be in the past tense.

3.3 Message Creation

We need to map each identified interaction to one or more messages in the output MSC. We also need to identify the sender (initiator of the interaction) and receiver (other actors involved in the interaction) for each message. We have developed several approaches for identifying a set of senders (SX) and a set of receivers (RX) for each valid interaction verb. If SX and RX are both empty, we ignore that interaction. If only one of them is empty, we add a special actor *Environment* (ENV) to that set. Once such sets are identified, a message is created for each unique combination of a sender and a receiver for a particular interaction verb.

Dependency parsing-based Approaches: We developed two approaches for message creation based on dependency parsing output: i) Baseline **B1** which directly maps the dependencies output to messages and ii) Approach **M1** (Algorithm 1) which builds on the dependencies output by applying additional linguistic knowledge. We use Stanford CoreNLP (Manning et al., 2014) for dependency parsing.

Baseline B1 simply maps each interaction verb in the dependency tree to a set of messages. Actors directly connected to an interaction verb with certain dependency relations ($nsubj$, $nmod{:}agent$) are identified as senders whereas actors directly connected to the verb with certain other dependency relations ($dobj$, $nsubjpass$, $xcomp$, $iobj$, $advcl{:}to$, $nmod{:}*$) are identified as receivers.

Approach M1 improves upon this baseline by generalizing connections between the verb and potential senders and receivers. Rather than considering only direct connections in dependency tree, M1 identifies certain actors as senders which are connected to the verb with a set of allowable dependency paths such as $nmod{:}poss \rightarrow nsubj$ or $nsubj \rightarrow advcl$ (lines 3-9 in Algorithm 1). E.g., consider the sentence `Bravery of Rajputs pushed the Mughals back.` Here, `Rajputs` is not directly connected to `pushed` in the dependency tree. Still, M1 would be able to identify `Rajputs` as sender because its dependency path to the verb `pushed` is $nmod{:}of \rightarrow nsubj$. Similarly, M1 identifies certain actors as receivers which are descendants of the verb in the dependency tree and the dependency paths connecting them to the verb satisfy certain properties such as "no other verb is allowed on the path" (lines 10-13 in Algorithm 1). Presence of another intermediate verb on such dependency path is a strong indicator that the receiver is an argument for the intermediate verb. For example, in the sentence `Crossing the Alps, Napoleon attacked Italy.`, "`the Alps`" is not a valid receiver for the verb `attacked` because another verb `Crossing` occurs on the dependency path connecting `the Alps` to `attacked`.

SRL-based Approaches: We developed two approaches for message creation based on SRL: i) Baseline **B2** which directly maps the SRL output to messages and ii) Approach **M2** (Algorithm 2) which builds on the SRL output by applying additional linguistic knowledge. We use MatePlus (Roth and Lapata, 2015) for SRL which produces predicate-argument structures as per PropBank (Kingsbury and Palmer, 2002). The baseline B2 simply maps each verbal predicate corresponding to an interaction verb to a set of messages. Actors corresponding to A0 arguments of a verbal predicate are identified as senders whereas actors corresponding to other arguments are identified as receivers.

Approach M2 improves upon this baseline by using VerbNet (Schuler, 2005) roles (the function $vnrole$) associated with PropBank arguments. Certain selectional preferences are used on these VerbNet roles, so as to qualify them as valid senders or receivers. These preferences are based on the linguistic knowledge and the details are described in the Algorithm 2. E.g., consider the sentence `Peter described John as very polite.` Here, for the communication verb `describe`, $vnrole$ (describe.01.A1) $= theme$. As per our linguistic rule, even if any actor is part of $theme$ of a communication verb, that actor does not qualify to be a receiver, as it is not directly participating in the interaction. Line 18 in Algorithm 2 encodes this rule, thereby not allowing any actor which is part of a $theme$ to be a receiver. Hence, in this example sentence, `John` will not be a receiver for `describe`.

Algorithm 2 also handles a special case about *Ergative* verbs which lie in between the spectrum of transitive and intransitive verbs. Their most distinguishing property is that when an ergative verb does not have a direct grammatical object, its grammatical subject plays an object-like role. E.g., consider following two sentences containing an ergative verb `move`:
S1: `Mao's father moved him to a hostel.`
S2: `Mao moved to Beijing.`
In S1, `moved` has an object but in S2, it does not have any direct object. Semantic Role Labelling would assign the role A1 (thing moving) to `Mao` in S2 and hence it can not be a sender. But as S2 indicates that the actor (`Mao`) is willingly performing the action of moving, we expect `Mao` to be a sender. Hence, for an ergative verb, even if the SRL assigns A1 role to an actor, we consider such an actor for being sender if no A0 role is assigned for the ergative verb by the SRL (lines 9-13 in Algorithm 2).

Combined SRL and Dependency parsing based Approach (M3): SRL tools are useful to identify senders and receivers of a message, but they do have a few important limitations. E.g. (i) SRL tool may fail to identify any A0 even when it is present or when it assumes the verb does not require A0 in the sentence; (ii) the identified A0 may be wrong or cannot be considered as a sender; (iii) SRL tool may fail to identify any A1/A2 even when it is present; (iv) the identified A1/A2 may be wrong or cannot be considered as a receiver.

We call this combined approach as **M3** which corrects the output of SRL-based approach M2 by using output of the dependencies based approach B1. The intuition, here is that B1 uses only high-precision rules for identifying senders and receivers. Hence, B1's output can be used to correct a few errors introduced in the M2's output. E.g., in `He was accorded a very`

Algorithm 1: $create\_messages\_M1$

input : s (sentence), A (set of known actors with coreferents), v (interaction verb), $DPOSS = \{nmod:poss, nmod:of\}$, $DS = \{nsubj, nmod:agent\}$, $DR = \{dobj, iobj, nmod*, xcomp, nsubjpass, advcl{:}to\}$, $DI = \{advcl{:}to, xcomp\}$

output : SX = set of senders, RX = set of receivers

$SX, RX := \emptyset$

$E_d := GetDependencyTree(s)$

// E_d is set of tuples of the form (a, b, dr) where a is governor of b with dependency relation dr

foreach *actor* $a \in A$ *s.t.* a *has mention in* s **do**

  if $(v, a, ds) \in E_d \ \wedge \ ds \in DS$ **then** $SX := SX \cup \{a\}$; **continue**

  if $(v, a, nmod{:}*with) \in E_d$ **then** $SX := SX \cup \{a\}$; **continue**

  if $\exists u \ s.t. (u, v, advcl*) \in E_d \wedge (u, a, ds) \in E_d \ \wedge \ ds \in DS$ **then** $SX := SX \cup \{a\}$; **continue**

  if $\exists u \ s.t. (v, u, ds) \in E_d \ \wedge \ ds \in DS \ \wedge \ u.ner = OTHER \ \wedge \ (u, a, dp) \in E_d \ \wedge \ dp \in DPOSS$ **then** $SX := SX \cup \{a\}$; **continue**

  if $\exists b \ s.t. \ b \in SX \ \wedge \ (b, a, nmod{:}*with) \in E_d$ **then** $SX := SX \cup \{a\}$; **continue**

foreach *actor* $b \in A \setminus SX$*s.t.* b *has a mention in* s *and* $\exists$ *path* P *from* v *to* b *in* G *using* E_d **do**

  if $\exists u \neq v \ s.t. \ u.POS = VB* \ \wedge u \in P \ \wedge \ (v, u, dr) \in E_d \ \wedge \ dr \notin DI$ **then continue**

  if $\exists u \neq v \ s.t. \ u.POS = VB* \ \wedge u \in P \ \wedge \ (v, u, *) \notin E_d$ **then continue**

  if $\exists x$ *in* $P s.t. (x, b, dr) \in Ed \ \wedge \ dr \in DR$ **then** $RX := RX \cup \{b\}$

return (SX, RX)

Algorithm 2: $create\_messages\_M2$

input : s (sentence), A (set of known actors with coreferents), v (interaction verb), $B_0 = \{agent, theme, cause\}$, $B_1 = \{experiencer\}$, $B_2 = \{AMLOC, AMDIR\}$, $B_3 = \{asset, cause, extent, instrument, stimulus, time, topic, theme, predicate\}$, $B_4 = \{theme\}$, $B_5 = \{agent, theme\}$

output: SX = set of senders, RX = set of receivers

$H := MatePlus(S)$; // output of MatePlus

$SX, RX := \emptyset$;

if $v \notin H \vee$ *is_copula_like(v)* **then return**(SX, RX)

if $H.v$ *has argument* A_0 **then**

  $x := H.v.A_0.phrase$;

  if x *contains an actor from* A **then**

    if $vnrole(H.v.A_0) \in B_0 \vee (is\_comm(H.v) \wedge vnrole(H.v.A_0) \in B_1)$ **then** $SX := SX \cup \{get\_actor(x, A)\}$;

else if $is\_ergative(v) \wedge H.v$ *has argument* A_1 **then**

  $x := H.v.A_1.phrase$;

  if x *contains an actor from* A **then**

    if $vnrole(H.v.A_1) \in B_5$ **then** $SX := SX \cup \{get\_actor(x, A)\}$;

foreach *argument* A_i $(i > 0)$ *in* $H.v$ **do**

  $x := H.v.A_i.phrase$;

  if x *contains no actor from* $A \setminus SX$ **then continue**

  if $H.v.A_i \in B_2 \vee vnrole(H.v.A_i) \notin B_3 \vee (is\_action(H.v) \wedge H.v.A_i \in B_4)$ **then** $RX := RX \cup \{get\_actor(x, A)\}$;

if $H.v$ *has another predicate* v' *as argument* **then**

  $SX', RX' := create\_messages\_M2(S, A, v')$;

  $RX := RX \cup RX'$;

return(SX, RX)

`cordial reception and was loaded with gifts.`, MatePlus (in M2) identifies `He` as A0 for `accorded`, which is wrong because `He` is not the initiator of this interation; `He` should be A1 for `accord`. We correct this by using the fact that B1 (dependencies based approach) detects the **nsubjpass** dependency between `accorded` and `He` and identifies `He` as receiver. As another example, for `His father united him in an arranged marriage to Luo Yigu, thereby uniting their land-owning families.`, MatePlus does not identify any A0 for `uniting`, where the true A0 is `His father`, which we correct using the dependency parse in which `His father` is connected to `uniting` through the path $nsubj \rightarrow advcl$.

3.4 Message Label Generation

We propose a simple algorithm for generating a clear and intuitive label for each message, covering various scenarios. For a verbal event, the label includes the main verb (`joined`), followed by a particle if present (`set_up`), a preposition if present (`cut_off_from`), a negation if present (`not_cut_off_from`), a secondary verb if present (infinitive, gerund or past participle), which also may be followed by a particle and/or preposition (`set_up_to_defend`, `helped_organize`, `set_up_for_taking_away_from`). The general syntax of our message label is given by the regex: `NEG? MAIN_VERB PARTICLE? (PREP|to)? (NEG? SECONDARY_VERB PARTICLE? PREP?)?`. We do not include adverbs, nor any nominal objects and arguments as part of the message label. We also do not include any auxiliary or modal verbs; e.g., from `had fled`, `was elected` we get the message labels `fled`, `elected`. Syntactic verbal structures such as `could have helped` indicate interactions that may not have taken place; so no messages are created for them.

3.5 Temporal Ordering of Messages

Temporal ordering of messages in a MSC is the final step and an important sub-problem of the over-

all high-level goal of automated MSC extraction. To order the messages, it is important to assign a temporal anchor to each message. A temporal anchor is a point in time (such as `1795-10-01`), at which an interaction has happened. The granularity of the temporal anchor is defined at the level of a year (`1795`), a month (`1795-10`) or a day (`1795-10-01`), but not lower.

We can observe sentences in a narrative which contain explicit time expressions (timex). Explicit timex are date points which are self-contained (e.g., `October 1795`) or can be resolved based on previously occurring dates (e.g., `Three months earlier`). Temporal anchors of messages in such sentences can be assigned normalized values of the explicit timex. To achieve this, we first identify these explicit timex and normalize them using the Heideltime timex recognizer and normalization system (Strötgen and Gertz, 2015). Secondly, the normalized explicit timex is assigned as the temporal anchor of the message which is present in the sentence. In case of sentences with multiple message verbs, the normalized explicit timex is assigned as the temporal anchor of the message which has its main verb nearest to the timex in the sentence's dependency tree.

However, it is important to note that messages may be in sentences without any explicit timex. In order to find the temporal anchor of such messages we employ the "document level time-anchoring (DLT)" algorithm proposed by (Laparra et al., 2015). The algorithm takes a list of messages (as per the text order) and document creation time (DCT) as inputs. The key assumption behind the algorithm is that all the messages of exactly same tense tend to occur in the text order, unless stated explicitly. In other words, the author will mention an explicit timex for the current message with tense t, only if its temporal anchor is different from the anchor of the last message of tense t.

The algorithm proceeds as follows: If a message m has a time anchor t obtained from an explicit timex, then t is stored in a tense-to-anchor map as the last seen anchor associated with the tense of m. However, if m does not have a temporal anchor assigned, then the last seen anchor of the message's tense is obtained from the tense-to-anchor map and set as m's temporal anchor. If the tense-to-anchor map does not have a mapping for m's tense then the provided DCT is set as m's temporal anchor.

Once all messages are assigned some temporal anchor, a simple sorting algorithm is used to order the messages based on their anchors. While sorting it is taken care that the assumption of *ordering messages with the same temporal anchor by their text order* is maintained.

4 Experimental Evaluation

4.1 Datasets

We evaluate our approach on history narratives as they are replete with multiple actors, spatio-temporal details and have varied forms of interactions. We choose public narratives of varying linguistic complexity to cover a spectrum of history: (i) famous personalities: Napoleon (**Nap**) (Littel, 2008), and Mao Zedong (**Mao**) (Wikipedia, 2018), (ii) a key event: Battle of Haldighati (**BoH**) (Chandra, 2007), and (iii) a major phenomenon: Fascism (**Fas**) (Littel, 2008).

We also use a subset of the Facebook's bAbI QA dataset (Weston et al., 2015) which is a text understanding and reasoning benchmark. Our **bAbI** dataset includes 10 instances from the time-reasoning subset of the bAbI QA dataset. Each instance consists of two interleaved sets of information: a set of sentences describing an event and its time for e.g. `Mary went to the cinema yesterday.`, and a set of temporal reasoning questions which need to be answered based on the sentences seen till that instant. We remove the questions from each instance keeping only the event description sentences as input to the approach.

We manually annotated these datasets for independent actor mentions, their aliases (canonical mentions), interaction verbs, complete messages and temporal ordering of the messages. Number of sentences and messages for the datasets are: Nap (106, 99), Fas (117, 115), BoH (77, 133), Mao (58, 135) and bAbI (118, 118).

4.2 Evaluation

We give highest priority to the message label and hence senders / receivers of a message are deemed to be correct only if the corresponding message label has been identified correctly. As one of the evaluation measures, we report the F-measure for identifying only the message labels, ignoring the corresponding senders / receivers.

We further evaluate message identification performance of the proposed approaches at two levels: i) complete messages with actor mentions

(denoted as L_1 level) and ii) complete messages with canonical mentions of the actors (L_2 level). As described in Section 3.1, each actor mention has a canonical mention associated with it, which represents a group of corefering actor mentions. At L_1 level, a predicted message is counted as a true positive if the combination of the predicted sender mention, receiver mention and message label (i.e., the complete message) is present in the gold-standard messages for the same sentence. False positives and false negatives are computed on similar lines and overall F-measures are computed for identifying complete messages, at the actor mention level. Similarly, the corresponding F-measures at L_2 (canonical mention) level are also computed by considering canonical senders / receivers instead of their mentions.

We conduct the experiments in two different settings: i) Setting S_1: using gold-standard information about actor mentions, canonical mentions and interaction verbs ii) Setting S_2: using predicted actors and interaction verbs. We use the approach proposed by Patil et al. (2018) for predicting actor mentions and identifying canonical mentions; and a simple algorithm for predicting interaction verbs. For evaluating our temporal ordering approach, we use Kendall's τ rank correlation coefficient (Kendall, 1938) to compare predicted and gold time-lines of a key actor in each dataset (e.g., `Mao Zedong` in the **Mao** dataset).

As goal of Kof's work (Kof, 2007) is same as our work on message extraction, we use it as one of the baselines (B-Kof). We also use OpenIE (Mausam et al., 2012) as another baseline (B-OIE). To avoid unnecessarily penalizing B-OIE, we consider only those extractions where relations fit our definition of interaction verbs and arguments fit our definition of actors. We compare our temporal ordering approach with the default text order based baseline (Text-Order). Table 2 shows comparative performance of the proposed approaches for message extraction and temporal ordering.

4.3 Analysis of Results

It can be observed in Table 2 that our proposed approaches M1 and M2 are consistently outperforming their corresponding baselines for all datasets in Setting S_1. Also, the approach M3 outperforms all other approaches in Setting S_1 when considering actor mentions for the complete message.

F1-measures in the setting S_2 get reduced considerably as compared to S_1. Our approach is a pipeline-based approach where output of actor and interaction verb identification are provided as input for the message creation algorithms. So, the errors in these earlier stages are propagated to the message creation stage, resulting in lower performance for the overall pipeline. Especially, identifying coreferences of actor mentions to determine canonical mentions, is a hard problem (Ng, 2017). Hence, in the setting S_2, we see a significant drop in F1-measure when we go from L_1 level messages to the L_2 level where identification of correct canonical sender / receiver is important.

History narratives tend to describe interactions mostly in the order in which they happen. Hence, we can observe that performance of the DLT based approach and Text-Order baseline is almost similar for the History datasets. In some instances, DLT based approach performs poorly as the default fall back for any previously unobserved tense is the DCT. This can be incorrect if a message with its verb in past_participle is anchored at DCT even after observing multiple previous messages in past tense anchored at an earlier time point. For datasets like bAbI in which text order of interactions differs significantly from the actual temporal order, the performance of the DLT based temporal ordering approach is better than the baseline.

5 Related Work

Though there has been some work in applying MSC for Software Engineering domain, less attention is given to the automatic construction of MSC using NLP. Feijs (2000) proposed an "object-oriented" approach to automatically construct an MSC from a narrative. The approach makes use of a set of generative rules in the form of a grammar. Kof (2007) proposed an approach to construct MSC for modelling scenarios from requirement analysis documents. Kof's approach is based on the situation stack based notion of human attention in a discourse (Grosz et al., 1995). However, the approach makes naive assumptions while finding senders, receivers and action verbs. For example, a sentence contains only one action verb, actors can be found in a pre-defined list and so on. As history narratives include multiple senders/receivers/action verbs and the actors are not pre-specified in a sentence Kof's approach (Kof, 2007) is less suitable.

Our work is close to the work by Chambers and

Dataset	Approach	Message Label		Complete Message				Temporal Ordering			
				Actor Mentions		Canonical Mentions		Text-Order		DLT	
		S_1	S_2	S_1	S_2	S_1	S_2	S_1	S_2	S_1	S_2
Nap	B-OIE	0.54	0.42	0.38	0.18	0.38	0.18	-	-	-	-
	B-Kof	0.32	0.25	0.17	0.08	0.17	0.08	-	-	-	-
	B1	0.92	0.67	0.49	0.28	0.49	0.32	-	-	-	-
	B2	0.94	0.70	0.64	**0.32**	0.62	**0.34**	-	-	-	-
	M1	**0.95**	0.68	0.51	0.26	0.51	0.31	0.99	0.99	0.95	0.99
	M2	0.94	**0.71**	0.65	0.29	**0.64**	0.29	0.99	0.99	0.99	0.99
	M3	0.94	**0.71**	**0.66**	**0.32**	**0.64**	0.33	0.99	0.99	0.93	0.93
Fas	B-OIE	0.56	0.51	0.44	0.28	0.43	0.19	-	-	-	-
	B-Kof	0.41	0.29	0.22	0.12	0.22	0.07	-	-	-	-
	B1	0.93	**0.63**	0.58	0.31	0.58	0.25	-	-	-	-
	B2	0.92	0.62	0.59	0.29	0.59	0.22	-	-	-	-
	M1	**0.94**	0.60	0.59	0.29	0.59	0.23	0.99	1.0	0.96	0.9
	M2	0.92	**0.63**	0.64	0.28	0.64	0.22	0.99	0.99	0.96	0.89
	M3	0.92	**0.63**	**0.69**	**0.33**	**0.69**	**0.26**	0.97	0.99	0.94	0.89
Mao	B-OIE	0.48	0.50	0.34	0.24	0.35	0.19	-	-	-	-
	B-Kof	0.28	0.29	0.12	0.07	0.12	0.07	-	-	-	-
	B1	0.86	0.72	0.40	0.31	0.41	0.21	-	-	-	-
	B2	**0.93**	0.74	0.61	0.31	0.63	0.18	-	-	-	-
	M1	**0.93**	**0.76**	0.44	0.31	0.45	**0.22**	0.88	0.88	0.84	0.84
	M2	**0.93**	0.73	**0.65**	**0.34**	**0.67**	0.20	0.90	0.88	0.86	0.88
	M3	**0.93**	0.73	**0.65**	0.33	0.66	0.21	0.90	0.88	0.86	0.88
BoH	B-OIE	0.39	0.40	0.28	0.19	0.28	0.04	-	-	-	-
	B-Kof	0.25	0.22	0.09	0.06	0.09	0.02	-	-	-	-
	B1	0.91	0.79	0.58	0.50	0.51	0.21	-	-	-	-
	B2	**0.96**	0.80	0.63	0.43	0.59	0.21	-	-	-	-
	M1	**0.96**	**0.81**	0.64	0.47	0.56	**0.22**	0.96	0.96	0.84	0.81
	M2	**0.96**	0.79	0.65	0.46	0.61	0.21	0.96	0.96	0.84	0.81
	M3	**0.96**	0.79	**0.71**	**0.52**	**0.65**	**0.22**	0.96	0.96	0.84	0.81
bAbI	B-OIE	**1.00**	**1.00**	**1.00**	**0.81**	**1.00**	**0.81**	-	-	-	-
	B-Kof	0.83	0.67	0.83	0.67	0.83	0.67	-	-	-	-
	B1	**1.00**	**1.00**	0.95	0.77	0.95	0.77	-	-	-	-
	B2	**1.00**	**1.00**	0.46	0.39	0.46	0.39	-	-	-	-
	M1	**1.00**	**1.00**	**1.00**	**0.81**	**1.00**	**0.81**	0.73	0.73	1.0	1.0
	M2	**1.00**	**1.00**	**1.00**	**0.81**	**1.00**	**0.81**	0.73	0.73	1.0	1.0
	M3	**1.00**	**1.00**	**1.00**	**0.81**	**1.00**	**0.81**	0.73	0.73	1.0	1.0

Table 2: F1-measures for following approaches- B-OIE: OpenIE baseline, B-Kof: Kof (2007), B1: Baseline using only dependencies, B2: Baseline using only SRL, M1: $create\_messages\_M1$ (Algorithm 1), M2: $create\_messages\_M2$ (Algorithm 2), M3: Combined approach using SRL and dependencies. Setting S_1 corresponds to using gold actors and interaction verbs, Setting S_2 uses predicted actors and interaction verbs

Jurafsky (2009) on modelling of narrative schemas and their participants. They need a *corpus of narratives* to identify prototypical schemas which try to capture common sequence of events. We address a different problem of extracting MSC from a *single* narrative and do not need a corpus. MSC has been proposed as a knowledge representation for a narrative text in (Bedi et al., 2017). We extend their work to automatically construct MSC.

Open Information Extraction (OpenIE) systems aim to extract tuples consisting of relation phrases and their multiple associated argument phrases from an input sentence (Mausam et al., 2012). The predicate-argument structures in OpenIE seem similar to SRL and dependency parsing. However, in dependency parsing the relations are fixed, while SRL systems require deeper semantic analysis of a sentence and hence they depend on lexical resources like PropBank and FrameNet. On the other hand, the predicate-argument structures in OpenIE are not restricted to any pre-specified or fixed list of relations and arguments.

6 Conclusions

Message Sequence Charts (MSC) is an important knowledge representation to summarize and visualize narratives such as historical texts. We proposed algorithms to automatically extract MSC from history narratives. We observed that the state-of-the-art systems of dependency parsing and SRL can not be used as-is for the task. Combining dependency parsing, SRL and linguistic knowledge achieves the best performance on different narratives. We also report results on temporal ordering of messages in the MSC using a tense based temporal anchoring approach.

References

Rajeev Alur, Gerard J Holzmann, and Doron Peled. 1996. An analyzer for message sequence charts. In *International Workshop on Tools and Algorithms for the Construction and Analysis of Systems*, pages 35–48. Springer.

Harsimran Bedi, Sangameshwar Patil, Swapnil Hingmire, and Girish K. Palshikar. 2017. Event Timeline Generation from History Textbooks. In *NLP-TEA@IJCNLP*, pages 69–77.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, pages 602–610.

Satish Chandra. 2007. *Medieval India: From Sultanat to the Mughals- Mughal Empire: Part Two*. Har Anand Publications.

Quang Do, Wei Lu, and Dan Roth. 2012. Joint inference for event timeline construction. In *EMNLP-CoNLL*, pages 677–687.

Loe M. G. Feijs. 2000. Natural language and message sequence chart representation of use cases. *Information & Software Technology*, 42(9):633–647.

Amit Goyal, Ellen Riloff, and Hal Daumé III. 2010. Automatically Producing Plot Unit Representations for Narrative Text. In *EMNLP*, pages 77–86.

Barbara J. Grosz, Scott Weinstein, and Aravind K. Joshi. 1995. Centering: A framework for modeling the local coherence of discourse. *Comput. Linguist.*, 21(2):203–225.

Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Nam Wook Kim, Benjamin Bach, Hyejin Im, Sasha Schriber, Markus H. Gross, and Hanspeter Pfister. 2018. Visualizing Nonlinear Narratives with Story Curves. *IEEE Trans. Vis. Comput. Graph.*, 24(1):595–604.

Paul Kingsbury and Martha Palmer. 2002. From treebank to propbank. In *LREC*, pages 1989–1993.

Leonid Kof. 2007. Scenarios: Identifying Missing Objects and Actions by Means of Computational Linguistics. In *15th IEEE International Requirements Engineering Conference (RE 2007)*, pages 121–130.

Egoitz Laparra, Itziar Aldabe, and German Rigau. 2015. Document level time-anchoring for timeline extraction. In *ACL (2)*, pages 358–364.

Liwu Li. 2000. Translating use cases to sequence diagrams. In *Proceedings of the 15th IEEE International Conference on Automated Software Engineering (ASE'00)*, page 293.

McDougal Littel. 2008. *World History: Patterns of Interaction*. World History: Patterns of Int. Houghton Mifflin Harcourt Publishing Company.

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

Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. 2012. Open language learning for information extraction. In *EMNLP-CoNLL*, pages 523–534.

Vincent Ng. 2017. Machine learning for entity coreference resolution: A retrospective look at two decades of research. In *AAAI*, pages 4877–4884.

Sangameshwar Patil, Sachin Pawar, Swapnil Hingmire, Girish K. Palshikar, Vasudeva Varma, and Pushpak Bhattacharyya. 2018. Identification of Alias Links among Participants in Narratives. In *ACL*.

Michael Roth and Mirella Lapata. 2015. Context-aware frame-semantic role labeling. *Transactions of the Association for Computational Linguistics*, 3:449–460.

Karin Kipper Schuler. 2005. Verbnet: A broad-coverage, comprehensive verb lexicon.

Jannik Strötgen and Michael Gertz. 2015. A baseline temporal tagger for all languages. In *EMNLP*, pages 541–547.

Piek Vossen, Tommaso Caselli, and Yiota Kontzopoulou. 2015. Storylines for structuring massive streams of news. In *Proceedings of the First Workshop on Computing News Storylines*, pages 40–49.

Piek Vossen, Tommaso Caselli, and Yiota Kontzopoulou. 2016. Storyline detection and tracking using dynamic latent dirichlet allocation. In *Proceedings of 2nd Workshop on Computing News Storylines*, pages 9–19.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*.

Wikipedia. 2018. Mao zedong — Wikipedia, the free encyclopedia. [Online; accessed 22-Feb-2018].