Paraphrase and Textual Entailment Generation

Zuzana Nevěřilová

NLP Centre, Faculty of Informatics,
Masaryk University, Botanická 68a,
602 00 Brno, Czech Republic
xpopelk@fi.muni.cz

Abstract. One particular information can be conveyed by many different sentences. This variety concerns the choice of vocabulary and style as well as the level of detail (from laconism or succinctness to total verbosity). Although verbosity in written texts is considered bad style, generated verbosity can help natural language processing (NLP) systems to fill in the implicit knowledge.

The paper presents a rule-based system for paraphrasing and textual entailment generation in Czech. The inner representation of the input text is transformed syntactically or lexically in order to produce two type of new sentences: paraphrases (sentences with similar meaning) and entailments (sentences that humans will infer from the input text). The transformations make use of several language resources as well as a natural language generation (NLG) subsystem.

The paraphrases and entailments are annotated by one or more annotators. So far, we annotated 3,321 paraphrases and textual entailments, from which 1,563 were judged correct (47.1 %), 1,238 (37.3 %) were judged incorrect entailments, and 520 (15.6 %) were judged non-sense.

Paraphrasing and textual entailment can be put into effect in chatbots, text summarization or question answering systems. The results can encourage application-driven creation of new language resources or improvement of the current ones.

Keywords: textual entailment, paraphrase, natural language generation.

1 Introduction

In human communication a lot of information is not mentioned, e.g. [1] observes that "[i]n human communication meaning is not conveyed by the text alone, but crucially relies on the inferential combination of the text with a context". The non-mentioned (implicit) information deserves attention since it is considerable part of communication in natural languages. [2, p. 149] estimates the ratio explicit:implicit information is up to 1:8.22 which means that the vast majority of information is to be inferred. Computer programs that simulate natural language understanding have to have this ability. This problem (also known as missing common sense knowledge) is well known and studied in artificial intelligence, cognitive science and linguistics.

When people explain something, they proceed in two ways: they can express the same thing in other words or they can explicitely voice the implicit knowledge. The former phenomenon is called *paraphrase*, the latter makes part of *textual entailment*.

P. Sojka et al. (Eds.): TSD 2014, LNAI 8655, pp. 293–300, 2014.

Textual entailment can be defined as a "relationship between a coherent text T and a language expression H, which is considered as a hypothesis. T *entails* H if the meaning of H, as interpreted in the context of T, can be deduced from the meaning of T" [3]. Entailment is often marked by the arrow symbol: $T \rightarrow H$.

Paraphrases are sentences with the same or almost the same meaning. Paraphrases can be seen as mutual entailments: if $T \rightarrow H$ and $H \rightarrow T$ then T and H are paraphrases.

In this work, we present a software that produces both entailments and paraphrases from one or more input sentences. By turning the implicit knowledge into explicit one, the system simulates natural language understanding. Paraphrasing and entailment generation can be used in question answering systems, text summarization, plagiarism detection, or chatbots.

The paper is organized as follows: in Section 2 we present works related to textual entailment recognition and generation, Section 3 describes the methods we use. Section 4 discusses the correctness of the generated sentences, in Section 5 we conclude our work and outline future tasks.

2 Related Work

2.1 Paraphrasing and Textual Entailment Datasets

Authors [4] distinguish paraphrase and textual entailment generation systems along another dimension: whether they recognize, generate or extract paraphrases or entailments. From this viewpoint, recognizing textual entailment (RTE) is the most studied topic. Eight workshops on RTE took place from 2004 to 2013, at first as the Pascal RTE challenges, then as tracks on Text Analysis Conference (TAC), and recently as a track on SemEval challenge[1]. All datasets from Pascal RTE challenges are available, so future RTE systems can undergo an evaluation using these benchmark datasets.

Independently on these RTE challenges, The Boeing-Princeton-ISI (BPI) Textual Entailment Test Suite[2] was developed. According to [5], it is syntactically simpler than Pascal RTE challenges but semantically more challenging. BPI focuses more on the knowledge than on linguistic requirements. The authors also classified the types of knowledge needed to successfully decide whether T entails H; moreover examples of these types are provided together with information about availability of the knowledge in Princeton WordNet (PWN) [6].

Microsoft Research Paraphrase Corpus (MSR) is the most widely used benchmark dataset for paraphrase recognition. From more than 5,000 pairs of sentences about two thirds are annotated as correct paraphrases.

Evaluation datasets do not exist only for English but also for Italian [7] and German [8]. The problem for non-English resources is the smaller diversity and size of language resources. For English language resources, the standardization and their aggregate use

[1] The up-to-date overview can be found in the ACLWiki `http://aclweb.org/aclwiki/index.php?title=Recognizing_Textual_Entailment`

[2] `http://www.cs.utexas.edu/~pclark/bpi-test-suite/`

is indoubtely the most developed. From the classification of knowledge types[3] made by [5], it is clear that the entailments and paraphrases can be recognized, generated or extracted only from big and manifold language resources.

2.2 Paraphrasing and Textual Entailment Applications

Authors [9] describe different RTE methods, from bag-of-words or vector space models to logic-based representations, syntactic interpretations, similarity measures on symbolic meaning representation, and decoding techniques. In the latter, rule-based transformations based on replacing synonyms, hyponyms by hypernyms, and application of paraphrasing patterns result in new sentences.

Some representants of existing RTE systems are:

- VENSES (Venice Semantic Evaluation System)[4]—a cross-platform system for RTE based on two subsystems: GETARUN (a system for text understanding) and the semantic evaluator initialy created for summary and question evaluation.
- EXCITEMENT Open Platform[5] is an open source platform for RTE. The system separates linguistics processing and entailments. It is pre-trained in three languages and it is further trainable. The software has an online demo. BIUTEE[6] (Bar Ilan University Textual Entailment Engine) was formerly a separate software, currently it is part of the EXCITEMENT project.
- EDITS (Edit Distance Textual Entailment Suite)[7] is a RTE software based on edit distance. It consists of three main modules: edit distance algorithm, cost scheme for edit operations, and a set of rules expressing either entailment or contradiction. EDITS works either in Italian or in English.
- Nutcracker[8] is a RTE system based on first order logic and theorem prover.

3 Methods

In this section, we describe our new system. Since it relies on different tools and language resources, we first describe them in short. Afterwards, we describe some paraphrasing and textual entailment generation tools. Each output sentence keeps information on the transformation type—we call this information a *signature*—which is helful in further analysis of the paraphrases and entailments.

[3] http://www.cs.utexas.edu/~pclark/bpi-test-suite/bpi-rte-knowledge-types.txt
[4] http://project.cgm.unive.it/venses.html
[5] http://hltfbk.github.io/Excitement-Open-Platform/
[6] http://u.cs.biu.ac.il/~nlp/downloads/biutee/protected-biutee.html
[7] http://edits.fbk.eu/
[8] http://svn.ask.it.usyd.edu.au/trac/candc/wiki/nutcracker

Table 1. Example LOSOP with the pronoun *jej* (*him*) replaced by the corresponding coreferent (*Jan Novák* or *John Doe*)

1: Jan Novák šel na procházku do temného lesa.		John Doe went for a walk in a dark forest.	
id: 1 *John Doe* word: Jan Novák lemma: Jan Novák tag: SG, NOM part: subject head: Jan constraint: +person	id: 3 *go* word: šel lemma: jít tag: SG, MASC, PAST, POSITIVE, 3RD PERS part: predicate	id: 5 *for a walk* word: procházku lemma: procházka tag: SG, ACC part: object constraint: -person ili: ENG20-00271999 prep: na	id: 7 *in the dark forest* word: temného lesa lemma: temný les tag: SG, GEN part: object head: les constraint: -person ili: ENG20-07926765 prep: do
2: Po setmění jej zachvátil šílený strach.		After the dusk a terrible panic seized him.	
id: 2 *after the dusk* word: setmění lemma: setmění tag: SG, LOC part: object head: setmění constraint: -person prep: po	id: 3 *John Doe* word: Jana Nováka lemma: Jan Novák tag: SG, ACC part: object constraint: +person dep: 4 coref: 1.1	id: 4 *seize* word: zachvátil lemma: zachvátit tag: SG, MASC, PAST, POSITIVE, 3RD PERS part: predicate	id: 5 *a terrible panic* word: šílený strach lemma: šílený strach tag: SG, NOM part: subject constraint: -person ili: ENG20-07058289 ENG20-07058791 …

3.1 Preprocessing

In our approach, we make use of several language resources that exist for Czech. The software is based on morphological analyser `majka` [10], tagger `desamb` [11], syntactic parser `SET` [12], and partial anaphora resolution tool Aara (yet unpublished). The input text is processed by these tools and then converted to its inner representation: we call it a list of set of phrases (LOSOP). An example LOSOP is shown in Table 1: the nodes are phrases (and not tokens as in a usual parse tree). Each phrase is annotated both syntactically (see the properties: *word*, *lemma*, *tag*[9], sentence *part*, *head*) and semantically (see the properties *constraint* and *ili*). We classify each subject or object according to the shallow ontology Sholva [13] (see the property *constraint*), and Czech WordNet [14] (see the property *ili*). So far, we only use two Sholva classes: `+person` and `-person`. Since we do not apply word sense disambiguation, each phrase is linked to all possible inter-lingual indices (ILI) of the Czech WordNet.

[9] We use the following abbreviations for the grammar categories: SG – singular, MASC – masculine, NOM – nominative, GEN – genitive, ACC – accusative, LOC – locative.

3.2 Synonym and Hypernym Replacement

Synonym replacement is one of the basic paraphrasing strategies. For synonym and hypernym replacement, we use Czech WordNet. The algorithm extracts maximum subphrases that exist as a literal in Czech WordNet. The subphrase must contain the head of the original phrase (e.g. *former minister of education* is a *minister* but not *education*). All adjective modifiers transform in order to preserve the grammatical agreement. For example, we can replace *auto* (neuter) by *vůz* (masculine, both meanint *the car*) and similarly *Janovo auto* to *Janův vůz* (*John's car*).

Hyponym replacement is very similiar to synonym replacement. Phrases are replaced by their hypernym, only if the predicate is positive (i.e. the head verb is not negative). Thus, *Peter came in his new convertible* can be transformed into *Peter came in his new car* but *Peter did not come in his new convertible* is not transformed.

As far as we do not employ word sense disambiguation, the result can be ambiguous as well. For example, the word *strach* can refer to `fear:1`, `worry:2`, or `panic:1` in PWN. The likelihood of a correct replacement is estimated upon the language model that computes the likelihood upon n-grams for $n = \{2, 3, 4, 5\}$ (further description can be found in [15]).

3.3 Verb Frame Equivalence

For entailments, we use the verb valency lexicon VerbaLex [16]. From VerbaLex, we extracted all synsets with phrase slots (i.e. no frames with idioms or subordinate clauses). We obtained 152,127 transformation rules that express verb frame equivalence (e.g. *to come* means the same as *to arrive*). In addition, we added 71 manual rules concerning not only equivalence but also preconditions and effects. An example rule can be seen in Figure 1. The rule can produce entailments such as *terrible panic seized* $X \rightarrow$ *X had fear.*

Verbs are more polysemous than nouns. Similarly to synonym and hypernym replacement, we do not disambiguate verbs, however, the verb frame syntactic structure is less ambiguous than the verb alone. We again rank the output sentences according to the language model mentioned in Section 3.2.

```
effect: zachvátit-bát se
 1:[type="predicate" lemma="zachvátit"] 2:[type="subject"
lemma="strach|panika|hrůza"] 3:[type="object" case="ACC"
constraint="+person"]
-> 1: [type="predicate" lemma="bát"] 3:[type="subject"]
5:[type="reflexive" lemma="se"]
```

Fig. 1. Verb frame inference rule that sets relation between *zachvátit* (*seize*) and (*strach|panika|hrůza* (*fear|panic|horror*), and *bát se* (*to have fear*)

3.4 Predicator Chains

[5] formulated 19 types of knowledge needed for successfull textual entailment. Our system covers at least three of them. However, three more could be covered by verb frame inference. The problem is the lack of language resources, although the phenomena of *predicator chains* is well-known and described by linguists.

Predicators are verb phrases in its functional relation to the clause. [17, p.32] distinguishes predicators on elementary and mutation predicators. Elementary predicators either describe a state or an elementary (impartible) process, mutation predicators indicate a change of state. Mutation predicators are semantically compound: a mutation predicator expresses a transition from state *a* to state *b*. State *a* is not explicitely mentioned but it is supposed to exist before the change takes place. For example the verb *zblednout* (to turn pale) implies that the subject was not pale before. Examples of each type of predicators is shown below:

- *process predicator* describes an action, e.g. *Matka suší prádlo* (Mother dries the laundry.)
- *mutation predicator* describes a process or state change, e.g. *Prádlo schne* (The laundry is drying.)
- *state predicators* describes a state, e.g. the predicator "to be" in sentences such as *Prádlo je suché* (The laundry is dry.)

The predicators of all three types form typical chains describing the states and their changes (here *to dry–to be drying–to be dry*). Computational linguists and cognitive scientists approach this phenomena in a complex manner by describing semantic frames (e.g. the FrameNet Project) or prototype theory. Nevertheless, this—purely linguistic—approach would cover many cases since predicator chains are often morphological (e.g. *to break–break–to be broken*).

Because the lack of a large language resource we manually crafted 34 rules (from those 71 mentioned in Section 3.3) that cover chains such as *to die–to be dead*, *set fire–burn–to be burnt*, *to sell–to be sold*. As we show in Section 4, the verb frame inference produces reasonable entailments.

4 Evaluation

The generated paraphrases and entailment were evaluated according to human judgment. For this purpose, we designed and implemented an annotation game presented in [15].

So far, we have collected 3,321 (non-unique) H–T pairs. From these pairs, 1,563 were judged correct (47.1 %), 1,238 (37.3%) were judged incorrect entailments, 520 (15.6 %) were on average judged non-sense. The game allows repeated annotations but the results show that players are not much motivated to annotate some previous text. Only 456 pairs were annotated more than once.

The overview of individual paraphrase and textual entailment generation methods is in Table 2. The method *no change* means that the input sentence was only analysed and generated from the corresponding LOSOP. *Other methods* comprise phrase reordering

and a possibility to enter a correct entailment manually. It can be seen that synonym replacement and verb frame equivalence are the less successful methods. Both methods work with ambiguous input and both produce a lot of noise. In future, we will concentrate on selecting the correct paraphrase rules. Currently, we implemented a ranking algorithm: the correlation between past annotations and particular signatures suggests the correctness of a paraphrase. For example, even though *kolo* can mean *wheel*, *round*, *lap*, or *bicycle*, the correlation between signatures *kolo*→ * and annotations shows that the dominating sense of *kolo* is *a bicycle*. For future paraphrases, the transformations of *kolo* in this sense will be preferred.

Table 2. Number of generated sentences per method

method	correct	incorrect	non-sense	% of correct	% of incorrect	total
no change	316	8	25	90.54	2.29	349
hypernym replacement	22	6	19	46.81	12.77	47
synonym replacement	434	854	173	29.71	58.45	1,461
anaphora resolution	276	127	95	55.42	25.5	498
verb frame equivalence	153	142	156	33.92	31.49	451
predicator chains	41	10	4	74.55	18.18	55
other methods	321	91	48	69.78	19.7	460
total	1,563	1,238	520	47.06	37.28	3,321

5 Conclusion and Future Work

This paper presents the generation phase of a bigger project. Although it is developed for Czech, the described techniques can be used for other languages as well. The only conditions are a variety and considerable size of the language resources. The system relies entirely on other NLP tools such as morphological analyser, tagger and syntactic parser. Their accuracy also affects the system's performance. Since the tools are in continuous development, our system's accuracy can increase in future.

There are many questions concerning the topic, for example: how one sentence can be generated from several input sentences? Currently, the system produces one sentence from one of the input sentences but it does not use wider context.

In future, we want to focus our research on two different directions: (1) error analysis (e.g. what annotators consider incorrect?) and (2) coverage estimation. It is clear that humans can produce paraphrases and entailments that are not covered by our system at all. In future, we want to identify more precisely those other types of paraphrases and entailments. We plan to extract paraphrases from texts describing the same topic.

This work can also encourage further development in language resources as it identifies clearly what types of language knowledge are missing.

Acknowledgments. This work has been partly supported by the Ministry of Education of CR within the LINDAT-Clarin project LM2010013 and by the Czech Science Foundation under the project P401/10/0792.

References

1. Gutt, E.A.: Implicit information in literary translation: A relevance-theoretic perspective. Target: International Journal of Translation Studies 8(2), 239–256 (1996)
2. Graesser, A.: Prose Comprehension Beyond the Word. Springer (1981)
3. Akhmatova, E.: Textual entailment resolution via atomic propositions. In: Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment (April 2005)
4. Androutsopoulos, I., Malakasiotis, P.: A survey of paraphrasing and textual entailment methods. CoRR abs/0912.3747 (2009)
5. Clark, P., Fellbaum, C., Hobbs, J.R.: The boeing-princeton-ISI (BPI) textual entailment test suite (December 2006), `http://www.cs.utexas.edu/~pclark/bpi-test-suite/` (accessed online April 14, 2014)
6. Fellbaum, C.: WordNet: An Electronic Lexical Database (Language, Speech, and Communication). The MIT Press (May 1998)
7. Pakray, P., Neogi, S., Bandyopadhyay, S., Gelbukh, A.: Recognizing textual entailment in non-English text via automatic translation into English. In: Batyrshin, I., Mendoza, M.G. (eds.) MICAI 2012, Part II. LNCS, vol. 7630, pp. 26–35. Springer, Heidelberg (2013)
8. Zeller, B., Padó, S.: A search task dataset for German textual entailment. In: Proceedings of the 10th International Conference on Computational Semantics (IWCS), Potsdam (2013)
9. Dagan, I., Roth, D., Zanzotto, F.M.: Tutorial notes. In: 5th Annual Meeting of the Association of Computational Linguistics. The Association of Computational Linguistics, Prague (2007)
10. Šmerk, P.: Towards Computational Morphological Analysis of Czech. Dissertation, Masaryk University in Brno (2010)
11. Šmerk, P.: K morfologické desambiguaci češtiny (Towards morphological disambiguation of Czech). Thesis proposal, Masaryk University (2008)
12. Kovář, V., Horák, A., Jakubíček, M.: Syntactic analysis using finite patterns: A new parsing system for Czech. In: Human Language Technology. Challenges for Computer Science and Linguistics, Poznań, Poland, November 6-8, Revised Selected Papers, pp. 161–171 (2011)
13. Grác, M.: Rapid Development of Language Resources. Dissertation, Masaryk University in Brno (2013)
14. Pala, K., Smrž, P.: Building Czech WordNet. Romanian Journal of Information Science and Technology 2004(7), 79–88 (2004)
15. Nevěřilová, Z.: Annotation game for textual entailment evaluation. In: Gelbukh, A. (ed.) CICLing 2014, Part I. LNCS, vol. 8403, pp. 340–350. Springer, Heidelberg (2014)
16. Hlaváčková, D., Horák, A.: VerbaLex – new comprehensive lexicon of verb valencies for Czech. In: Proceedings of the Slovko Conference (2005)
17. Grepl, M., Karlík, P.: Skladba spisovné češtiny. Edice Učebnice pro vysoké školy. Státní naklad (1986)