Master thesis project proposal

Automating Concept Alignment for Machine Translation

Arianna Masciolini (gusmasar@student.gu.se)

Suggested Supervisor at CSE: Aarne Ranta

Relevant completed courses:

(LT2214, Computational Syntax)
(LT2003, Natural Language Processing)
(DIT143, Functional Programming)
(DIT260, Advanced Functional Programming)
(DIT231, Programming Language Technology)
(DIT300, Compiler Construction)
(DIT411, Introduction to Artificial Intelligence)
(DIT866, Applied Machine Learning)

September 15, 2020

1 Introduction

Concept Alignment (CA) consists in finding semantical correspondences between parts of parallel texts in two or more different languages, of which the Rosetta Stone is a notable example. Such task, which is routinely performed by learners of classical languages, who generally work with a translation alongside the original text, is often preliminary to further linguistic analysis.

Another task that involves CA is natural language translation: the human translator, almost unconsciously, first identifies concepts in the source text and then looks for ways to render them in the target language. In this case, alignment can happen - and usually does happen simultaneously - at different levels of abstraction, ranging from word to sentence level. It is then natural to wonder wether it is possible to make use of CA in Machine Translation (MT) as well. The hypotheses motivating this project, whose objective is to develop and test strategies for automating CA, is that MT can benefit from CA in two ways:

- on the one hand, if we aim to design a *compositional* MT pipeline, CA is a necessary step, as it will become clearer in the following sections
- on the other hand, if we address the problem from the perspective of XMT (eXplainable MT), providing the user of a MT application with a way to compare pairs of concepts instead of the entire text in the source language to its automatically translated counterpart can help building a more easily interpretable, and therefore more reliable MT system.

2 Problem

To give a more rigorous definition of CA, it is first necessary to specify what we mean by *concept*.

2.1 Concepts

Intuitively, concepts are the components of meaning, and therefore, in a multilingual context, the units of translation. The principle of semantic compositionality states that the meaning of a complex expression is a function solely of the meanings of its components and of the manner in which these components are combined [5]. If we assume this principle to be valid and apply it to translation, these meaning components, which we refer to as concepts, are the common denominator between an expression in the source language and its translation, which can be generated using them as the starting point.

We can draw a very close parallel between MT and compiler pipelines: in this sense, concepts are the intermediate representation, or *abstract syntax*, obtained by analyzing, or *parsing*, the source language, while we will call *backend* the portion of the pipeline aimed at *generating* the target language by applying the concepts a set of rules.

In this perspective, then, concepts are abstract syntax functions. In the simplest case, when they correspond to individual words, their arity is zero, but it is often the case that the concrete expressions corresponding to minimal translation units are, in one or both of the source and target language, lemgrams, multiword expressions - sometimes discontinuous - or even more complex constructions, whose intermediate representation is an abstract syntax function taking one or more arguments.

2.2 Alignment

Having defined concepts more accurately, we can turn to the more concrete problem of CA, the task of finding, at any abstraction level, correspondences between parallel, and in the case of translation multilingual texts.

In order to facilitate its automation, CA can be divided into two subproblems subsequent to each other:

1. *concept creation*, the task of identifying concepts via language comparison, which is the main focus of this project

2. *concept propagation*, i.e. the task of finding, in a language which was not used in the concept creation stage, the concrete expressions corresponding to previously identified concepts.

3 Context

The idea of taking advantage of semantic compositionality is not new in the field of MT. Introduced by Curry in the early 1960s [2], it was first put in practice two decades later in the form of an experimental interlingual translation system, not coincidentally named Rosetta, which requires the definition of two distinct logically isomorphic Montague grammars - one for the source language, one for the target language - and constructs an intermediate representation based on such isomorphism [4].

3.1 Grammatical Framework

Grammatical Framework (GF), a grammar formalism and programming language designed with the parallel between compilers and MT systems discussed in the previous section in mind, goes a step further in this direction: it introduces a clear distinction between the abstract syntax, a phrase structure tree-like representation capturing the syntactic structures all natural languages taken into account have in common, and the *concrete syntaxes*, specific to each individual language, which consist in rules for the linearization of these Abstract Syntax Trees (ASTs) [7] [8]. This makes it possible to deal with multiple language by writing only one grammar, whose components are an abstract syntax and several concrete syntaxes.

GF grammars can be used in a variety of contexts, one of which is interlingual MT. More in particular, experiments with GF grammars have been conducted in the field of XMT, as the ASTs produced by source language analysis can serve both as to some extent automatically checkable certificates for the correctness of translation - by backlinearization to the source language - and as fully inspectable explanations aimed towards expert users [9].

When it comes to CA itself, which could potentially provide more user-friendly explanations, the idea of performing *syntax-based alignment* by comparing GF ASTs seems promising, especially if compared with the standard approaches used in statistical MT, like word alignment and its generalization, phrase alignment. Both approaches ignore sentence structure and therefore, while requiring a lot of data, their precision is low.

In practice, however, comparing GF ASTs presents a number of issues related to the fact that GF parsing is not robust enough to spelling and grammatical errors, unusual word orders and rare constructions in general [9]. It becomes then necessary, if we want to perform syntax-based alignment, to make use of an alternative parser. Preliminary experiments, not yet published, have been conducted exploiting the similarity between GF and UD (Universal Dependencies) [3, 10].

3.2 Universal Dependencies

UD is a framework for cross-linguistically consistent grammatical annotation. The UD project aims at developing parallel treebanks for many languages in order to support, among other things, the development of multilingual parsers.

Existing UD parsers, such as [6], are often neural pipelines trained on UD treebanks that, given raw text as input, output *dependency trees.* The difference between such trees and those produced by GF parsing is that dependency is a one-to-one correspondence, meaning that words in a sentence are connected to each other by directed links, while phrase structure is one-to-many.

If phrase structure trees seem to be a good starting point for natural (target) language generation, replacing them with dependency trees in the analysis step can prove to be a good basis for CA. In more concrete terms, if, as mentioned in the introduction, CA is seen as a step of compositional MT, a UD parser can be integrated in a GF-based pipeline, thus constructing a hybrid translation system, with a robust neural frontend and a phrase structure grammar-based backend. In this case, dependency trees need to be converted into GF ASTs, even though nlike its reverse [3], the UD-GF conversion is a non-deterministic search problem, addressed in [10].

4 Goals and Challenges

The overall goal of this project is to test and further develop the above mentioned ideas for CA automation.

A first part of the project consists in selecting an UD parser well suited to the task by manually evaluating the dependency trees obtained for a bilingual parallel text. Once an adequate parser is identified, the possibility of modifying such parser in order to tune it for the task at hand will be considered.

When it comes to alignment itself, which is our proposed way to address concept creation and as such the core part of this project, an unpublished algorithm for UD dependency trees alignment exists, but preliminary experiments show that there is room for improvement. The algorithm, in fact, that orders the trees and pads them to then collect matching subtrees, still has very low precision. Examples of specific problems to work on are making it so that it cab recover from parse errors and dealing with irrelevant cross-lingual differences between parse trees.

4.1 Stretch goals

If time allows it, a natural extension to this project is to start using more than just two languages, as result may differ significantly for different language pairs. Our criteria for selecting which languages to compare are described in the following section.

5 Approach

Building upon previous experiments, we will conduct a case study on the text of the General Data Protection Regulation (GDPR) [1]. As all European regulations, it is a highly multilingual parallel text, as it has a high quality translation for each of the 24 official languages of the European Union. Furthermore, for 5 of these languages (English, German, Spanish, Italian and French), a gold standard in terms of concept extraction, even if not publicly available, already exists.

Our choices in terms of which languages to use in this project are of course also limited by the linguistic knowledge of the candidate. As for the initial language pair to start with, we intend to use Italian and English, which, apart from being respectively her first and second language, belong to two different branches - resp. Romance and Germanic - of the Indo-European language family, making CA a potentially harder, but at the same time more interesting task. In the aforementioned case that we decide to also perform experiments with one or more other languages, it would be possible to work with yet another one for which we already have a gold standard available, Spanish, and/or we could consider using Swedish, which might involve a relatively demanding annotation process.

For obtaining dependency trees, the starting point will be one of the existing UD parsers yet to be chosen, while the syntactic alignment component will be based on an initial Haskell prototype.

5.1 Evaluation

For a first evaluation, the aforementioned manually annotated 5-lingual parallel text of the GDPR will suffice.

To assess the generalization capabilities of our model, however, it will be necessary to test it on other texts. As previously pointed out, the Official Journal of the European Union is itself a relatively large, highly multilingual parallel corpus of legal documents, but, even if good results within this context would already be somewhat meaningful, it would be interesting to put the system to the test in other domains, for instance using public domain literature.

References

[1] European Commission. General data protection regulation. *Official Journal of the European Union*, 2016.

[2] Haskell B Curry. Some logical aspects of grammatical structure. *Structure of language and its mathematical aspects*, 12:56–68, 1961.

[3] Prasanth Kolachina and Aarnte Ranta. From abstract syntax to universal dependencies. In *Linguistic Issues in Language Technology, Volume 13, 2016*, 2016.

[4] Jan Landsbergen. Machine translation based on logically isomorphic montague grammars. In *Coling 1982: Proceedings of the Ninth International Conference on Computational Linguistics*, 1982.

[5] Francis Jeffry Pelletier. The principle of semantic compositionality. *Topoi*, 13(1):11–24, 1994.

[6] Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D Manning. Universal dependency parsing from scratch. *arXiv preprint arXiv:1901.10457*, 2019.

[7] Aarne Ranta. Grammatical framework. *Journal of Functional Programming*, 14(2):145–189, 2004.

[8] Aarne Ranta. *Grammatical framework: Programming with multilingual grammars*, volume 173. CSLI Publications, Center for the Study of Language and Information Stanford, 2011.

[9] Aarne Ranta. Explainable machine translation with interlingual trees as certificates. In *Proceedings of the Conference on Logic and Machine Learning in Natural Language (LaML 2017)*, pages 63–78, 2017.

[10] Aarne Ranta and Prasanth Kolachina. From universal dependencies to abstract syntax. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 107–116, 2017.