Grammar-Based Concept Alignment for Domain-Specific Machine Translation

Anonymous ACL submission

Abstract

In grammar-based domain-specific MT, constructing high-quality translation lexica is a crucial but time consuming task requiring significant linguistic knowledge. With parallel example sentences available, statistical alignment tools can help automate part of the process, but they are not suitable for small datasets and limited when it comes to complex multiword expressions. In addition, the correspondences between word forms obtained in this way cannot be used directly. Addressing these problems, we propose a grammar-based approach to this task and put it to the test in a simple translation pipeline.

1 Introduction

Grammar-based translation pipelines such as those based on Grammatical Framework (GF) have been successfully employed in domain-specific Machine Translation (MT) (Ranta et al., 2020). What makes these systems well suited to the task is the fact that, when we constrain ourselves to a specific domain, where precision is often more important than coverage, they can provide strong guarantees in terms of grammatical correctness.

Nevertheless, lexical exactness is, in this context, just as important as grammaticality. An important part of the design of the Controlled Natural Language (CNL) one such system's grammar describes then becomes the creation of a high-quality translation lexicon.

This is often done manually, resulting in a time consuming task requiring significant linguistic knowledge. When the grammar is designed based on a corpus of parallel example sentences, it is possible to automate part of this process by means of statistical word and phrase alignment techniques (Brown et al., 1993; Och and Ney, 2000; Dyer et al., 2013). None of them is, however, suitable for the common case in which only a limited amount of example data is available.

In this paper, we propose an alternative approach to the automation of this task. While still being data-driven, our method is also grammar-based and, as such, capable of extracting meaningful correspondences even from individual sentence pairs.

A further advantage of performing syntactic analysis is that we do not have to choose *a priori* whether to focus on the word or phrase level. Instead, we can simultaneously operate at different levels of abstraction, extracting both single- and multiword, even non-contiguous, correspondences. For this reason, we refer to the task our system attempts to automate as *Concept Alignment* (CA).

Finally, performing morphological analysis allows us to generate translation lexica complete with grammatical category and inflection information instead of simple correspondences between word forms.

This paper is structured as follows. Section 2 starts by giving an overview of our approach to CA and comparing it to related work to then focus on our algorithm for extracting correspondences. After that, Section 3 presents the results obtained in a first evaluation of the system. Finally, Section 4 summarizes our conclusions and discusses some ideas for future work.

2 Methodology

The objective of CA is to find semantical correspondences between parts of multilingual parallel texts. We call *concepts* the abstract units of translation, composed of any number of words, identified through this process, and represent them as *alignments*, i.e. tuples of equivalent concrete expressions in different languages.

The basic use case for CA, which we refer to specifically as *Concept Extraction* (CE), is the generation of a translation lexicon from a multilingual parallel text. This can be directly compared to the numerous existing word and phrase alignment techniques.

An interesting and less studied variant of CA is *Concept Propagation* (CP), useful for cases where a set of concepts is already known and the goal is to identify the expressions corresponding to each of them in a new language, potentially even working with a different text in the same domain. While our system does implement basic CP functionalities, however, in this paper we focus on CE and restrict its application to bilingual corpora.

As stated in the Introduction, most existing extraction solutions are based on statistical approaches and are, as a consequence, unsuitable for small datasets. Grammar-based approaches, making use of parallel treebanks and collectively referred to as *tree-to-tree alignment methods*, have also been proposed (Tiedemann, 2011), but have historically suffered from the inconsistencies between the formalisms used to define the grammars for the various languages and from the lack of robustness of existing parsers. This work is a new attempt in the same direction, enabled by two multilinguality-oriented grammar formalisms developed over the course of the last 25 years: Grammatical Framework (GF) (Ranta et al., 2020) and Universal Dependencies (UD) (Rademaker and Tyers, 2019).

GF is a constituency grammar formalisms and programming language in which grammars are represented as pairs of an *abstract syntax*, playing the role of an interlingua, and a set of *concrete syntaxes* capturing the specificities of the various natural languages. In the case of translation, similarly to what happens in programming language compilation, strings in the source language are *parsed* to Abstract Syntax Trees (ASTs), which are then *linearized* to target language strings.

UD, on the other hand, is a dependency grammar formalism meant for cross-linguistically consistent grammatical annotation. As opposed to constituency, *dependency* is a word-to-word correspondence: each word is put in relation with the one it refers to, called its *head*, via a directed labelled link specifying the syntactic relation between them. Importantly for our application, the standard format for UD trees, CoNNL-U, gives information not only on the syntactic role of each word, but also on its Part-Of-Speech (POS) tag, lemma and morphological features.

While both formalisms independently solve the issues related to having to work with grammars that are inconsistent with each other, UD is especially appealing since, dependency trees being an easier target, several robust parsers, such as (Straka et al., 2016) and (Chen and Manning, 2014) are available. Alone, UD trees are sufficient to extract (or propagate) tree-to-tree alignments, but not to automate the generation of a morphologically-aware translation lexicon for a generative grammar. This is where GF comes into play: after correspondences are inferred from a parallel text, our system is able to convert them to GF grammar rules, easy to embed in a domain-specific grammar but also making it immediate to carry out small-scale translation experiments using pre-existing grammatical constructions implemented in GF's Resource Grammar Library (RGL), which covers the morphology and basic syntax of over 30 languages. This is enabled by gf-ud, a conversion tool described in (Kolachina and Ranta, 2016) and (Ranta and Kolachina, 2017). Concretely, then, the system we propose consists of a UD parser, an alignment module based on UD tree comparison and a program, based on gf-ud, that converts them into the rules of a GF translation lexicon.

2.1 Extracting concepts

The core part of the system outlined above is of course the alignment module. In this section, we present our method for extracting alignments from parallel bilingual UD treebanks. Our basic algorithm is sketched in th pseudocode below.

---

procedure EXTRACT(*criteria*,(t, u))
    alignments = $\emptyset$
    if (t, u) matches any alignment *criteria* **then**
        alignments += (t, u)
        for (t', u') in SORT(SUBTS(t)) $\times$ SORT(SUBTS(u))
    do
            extract(*criteria*,(t', u'))
    return *alignments*

Here, the input consists of a list of priority-sorted *alignment criteria*, i.e. rules to determine whether two dependency trees should be aligned with each other which will be discussed in greater detail in Section 2.1.1, and a pair of sentences to align represented as UD trees. From an implementation point of view, UD trees are, rose trees where each node represents a token, obtained from the CoNLL-U files produced by the UD parser.

As a first step, the program checks whether the two full sentence trees can be aligned with each other, i.e. if they match any of the alignment criteria. If this is the case, they are added to a collection of alignments, which are represented themselves
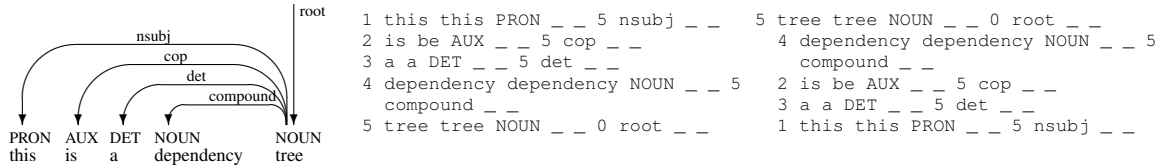
Figure 1: The graphical representation of a UD tree alongside its CoNNL-U and rose tree representation. Optional CoNNL-U fields are left blank.

as a pair of UD (sub)trees associated with some metadata, such as the id of the sentence they were extracted from. Such collection is what the function will return after aligning all the dependency subtrees. The same procedure is applied recursively to all pairs of immediate subtrees of each sentence, until the leaves are reached or alignment is no longer possible due to a lack of matching criteria. Subtrees are sorted based on their dependency label, so to give higher priority to pairs whose heads have the same label.

A simple but useful refinement is that, depending on which alignment criteria a pair of trees candidate for alignment matches, the heads of the two trees may or may not also be added to the collection of alignments. This is important not to miss one-word correspondences that cannot be captured in any other way, for instance between the root verbs of two full sentences. A relevant implementative detail is that, in this context, the head of a tree is not simply defined as its root. Instead, if the root is part of a compound written as two or more separate words or a verb with auxiliaries, the root nodes of the corresponding subtrees are also considered part of it.

When working on multiple sentences, this algorithm can be applied in an iterative fashion, so that knowledge gathered when a sentence pair is aligned can be reused when working on the following ones and to keep track of the number of occurrences of each alignment throughout the entire text.

2.1.1 Alignment criteria

While alignment criteria are customizable, to allow for a better understanding of the extraction algorithm described in the above, we describe the ones that the current implementation utilizes by default.

Matching UD labels The most obvious, but also most effective idea is to determine alignability based on comparing the dependency labels of the candidate UD tree pair. In particular, according to this idea, two subtrees in *matching context*, i.e. attached to aligned heads, constitute an alignment if their roots share the same dependency label, meaning that they are in the same syntactic relation with their heads. Note that, since the root of UD trees is always attached to a a fake node with an arc labelled `root`, this criterion also makes it so that full sentences are always considered to align with each other, something which is desirable since we assume the parallel texts that are fed to our program to be sentence-aligned.

Part-Of-Speech equivalence As noted in the above, CoNNL-U files also provide information on the grammatical categories of each word, represented as Universal POS tags (Petrov et al., 2012). Intuitively, if the nodes of two trees in matching contexts have the same POS tags, the two trees are more likely to correspond to each other than if not. This is especially true if we focus, for instance, solely on the open class words (defined as in the UD documentation [1]), thus ignoring prepositions, determiners, auxiliary verbs and all other function words, which tend to behave differently across different languages. A useful relation to define between dependency trees is, then, that of *POS-equivalence*: two dependency trees t, u are POS-equivalent if $M_1 = M_2 \neq \emptyset$, where M_i is defined as the multiset of POS tags of all the open class word nodes of t_i.

Applied as a backup for label matching, this criterion can be used to capture correspondences that would otherwise be missed, thus increasing recall, but a decrease in precision is also to be expected. However, since alignment criteria are defined as boolean functions, it is easy to combine them so to that they have to apply simultaneously. This can be useful in cases where precision is more important than recall.

Known translation divergence Parallel texts often present significant, systematic cross-linguistic grammatical distinctions. When this is the case, it is often straightforward to define alignment criteria

---

[1] universaldependencies.org/u/pos/all.html

3

based on recognizing the corresponding patterns. While many distinctions of this kind are specific to particular language pairs or even stylistical, some of them occur independently of what languages are involved and do not depend on idiomatic usage nor aspectual, discourse, domain or word knowledge. Drawing inspiration from (Dorr, 1994), we refer to them as *translation divergences* and handle some of the most common ones explicitly.

For instance, *categorial divergences* occur when the POS tag of a word in the source language changes in its translation. An ubiquitous example of is that of adverbs rendered as obliques, such as in the English-Italian pair ⟨*She studies* **consistently**, *Lei studia* **con costanza**⟩. *Structural divergences*, where a core argument of a sentence in the source language are rendered as obliques in the source language (as in the English-Swedish pair ⟨*I told* **him**, *Jag berättade* **för honom**.⟩), are also frequently encountered.

Known alignment Another case in which it is trivially desirable for two subtrees in matching context to be aligned is when an equivalent alignment is already known, for instance due to a previous iteration of the extraction algorithm. When referred to pairs of alignments, the term *equivalent* indicates that the two alignments, linearized, correspond to the same string.

At a first glance, this might seem a criterion with no practical applications. However, it is particularly useful when, instead of starting with an empty set of correspondences, we initialize the program with some alignments that are either inserted manually or, most interestingly, obtained with some other alignment technique. For instance, in this way it is possible to combine the system proposed in this paper in combination with a statistical tool and give more credit to the correspondences identified by both systems.

2.1.2 Pattern matching

So far, we described how CE can be used in a setting where the objective is to generate a comprehensive translation lexicon based on set of example sentence pairs. We pointed out that the program can be configured so to prioritize precision or coverage, but never restricted our search to a particular type of alignments. Nonetheless, there can be cases in which only certain syntactic structures are of interest: for instance, we might be looking for adverbs or noun phrases exclusively.

To handle these situations, the CE module can filter the results based on a `gf-ud` pattern. `gf-ud` supports in fact both simple pattern matching, which is integrated in the CE module itself, and pattern replacement[2]. Combining them, for instance by pruning the UD trees extracted by the alignment module, allows us to extract even correspondences that, given the structure of UD trees, cannot be identified by CE alone.

To give a practically useful example, a possibility is to extract verb phrases by looking for full clauses and pruning the subtrees corresponding to subjects. It is also possible to perform some actual replacement to obtain *predication patterns*, i.e. correspondences that specify the argument structure of verbs, such as the following English-Italian one:

⟨X *gives* Y Z, X *dà* Z *a* Y⟩.

2.2 Generating grammar rules

The CE module described so far outputs pairs of UD trees. While converting them to GF ASTs is one of `gf-ud`'s core functionalities, building a compilable GF translation lexicon is necessary to turn such trees into grammar rules.

In order to do that, our grammar generation module requires a morphological dictionary of the languages at hand and an *extraction grammar*. What we refer to as an extraction grammar is of course a GF grammar, whose objective is to define the set of basic categories and syntactic rules the entries of the automatically generated lexicon are going to use.

3 Evaluation

In this section, we evaluate the system proposed in the above. We first discuss the data used for such evaluation. After that, we describe our experiments, aimed at putting both the CE module *per se* and the entire system from parsing to lexicon generation to the test, and present our results.

3.1 Data

Because we want part of our evaluation to be independent from the quality of UD parsing, some of the experiments are carried out on treebanks instead of raw text. To this purpose, we use a 100-sentence subset of the Parallel UD (PUD) corpus, a set of 1000 manually annotated or validated sentences in CoNLL-U format. Of the over 20 languages PUD

not sure how important this section is and what (if anything) to add to it

---

[2]documentation is available at `github.com/GrammaticalFramework/gf-ud`

treebanks are available in, we selected English, Italian and Swedish. Using this data limits the amount of alignment errors that are due annotation issues to a minimum, even though a small number of inconsistencies and imprecisions is present even in this corpus.

When it comes to testing the program on raw text, we use two small (< 1000 sentences) bilingual sentence-aligned corpora consisting of course plans from the Department of Mathematics and Computer Science (DMI) of the University of Perugia (for English-Italian) and from the Department of Computer Science and Engineering (CSE) shared between the University of Gothenburg and the Chalmers University of Technology (for English-Swedish). For brevity, we will refer to these two datasets as to the DMI and the CSE corpora. This data, available in the project repository, was collected and sentence-aligned specifically for this work and a related Bachelor's thesis project (Eriksson et al., 2020). When using raw text, our parser of choice is UDPipe (Straka et al., 2016). In particular, we use the ParTUT English and Italian models for the DMI corpus and models trained on the bilingual LinES English-Swedish treebank for the CSE corpus.

3.2 Evaluating CE

While we focus mostly on the MT applications of CA, automatic translation, and much less GF-based domain-specific tranlation, is not the only context in which CA can be put to use. For instance, it is easy to imagine using it to improve the UI of online translation memories. For this reason, a first set of experiments is aimed at evaluating the alignments obtained with our CA module independently from the other stages stages of our lexicon generation pipeline.

We first assess the correctness of the alignments the CE module is able to extract from the PUD treebanks, comparing our results with those obtained with a standard statistical tool, `fast_align` (Dyer et al., 2013). In addition, we try to quantify the impact automatic parsing has on the performance of the CE module by comparing the above results with those obtained on the DMI and CSE corpora.

While precision and recall are two well-known performance metrics, the lack of a gold standard for CE imposes us to approximate them with the number of correct alignments the program is able to extract and the ratio between such number and the total number of alignments extracted. However, since some alignments are only correct in the specific context of the sentence pair in which they occur, we make a further distinction between correct alignments that are relevant for a translation lexicon and alignments that are useful for comparing the sentences but should not be used for MT. As an extreme example of a pair-specific alignment, consider the sentences ⟨*He missed the boat, Ha perso il treno*⟩. In English and Italian, the idea of missing a chance is expressed using two idiomatic expression very similar to each other. However, the vehicle mentioned in the Italian translation is not a boat but a train ("*treno*").

3.2.1 Results on manually annotated treebanks

In Table 1, we compare the results obtained with our grammar-based module to those obtained statistically on the PUD treebanks.

Of course, `fast_align` does not make use of the information present in the CoNLL-U files excepts with regards to tokenization. On the other hand, the relatively large size of the PUD treebanks makes it possible to also train the statistical tool on the full dataset instead of just using the chosen 100-sentences subset, allowing for a fairer comparison. In both cases, `fast_align` is run with the recommended parameters and the CE program is configured to only extract one-to-many and many-to-one word alignments, given that `fast_align` does not align larger phrases.

While `fast_align` is designed to align every word in the text (or explicitly state that a word has no counterpart in its translation) and, consequently, extracts around twice as many correspondences as our CE module, the percentage of correct correspondences is definitely in favor of our system, even though `fast_align` gets significantly more precise when trained on the full, 1000-sentence dataset.

3.2.2 Results on raw text

The course plan corpora are significantly harder to work with, the additional challenge being that our CE module relies on the quality of automatic parsing. A comparison between the results obtained on manually annotated data and the course plans corpora parsed with UDPipe is summarized Table 2.

What is immediately evident, but not unexpected,

	CE		fast_align (100 sentences)		fast_align (full dataset)	
	en-it	en-sv	en-it	en-sv	en-it	en-sv
distinct alignments	536	638	1242	1044	1286	1065
correct	392 (73%)	514 (80%)	346 (28%)	538 (52%)	540 (42%)	677 (64%)
usable in MT	363 (68%)	503 (79%)	316 (25%)	525 (50%)	510 (40%)	666 (63%)

Table 1: Comparison between our grammar-based CE module and `fast_align` on PUD data training the model on 100 and 1000 sentences and discarding the alignments obtained for sentences 101-1000 in the latter case.

	PUD (100 sentences)		course plans	
	en-it	en-sv	DMI (881 sentences)	CSE (539 sentences)
distinct alignments	1197	1325	1823	1950
correct	916 (77%)	1112 (85%)	1205 (66%)	1296 (66%)
usable in MT	880 (74%)	1099 (84%)	1157 (63%)	1248 (64%)

Table 2: Comparison the extraction of alignments of any size from manually annotated PUD treebanks and automatically parsed sentences from the course plans corpora.

is a decrease in precision. The percentage of correct alignments, however, stays significantly higher than that obtained with `fast_align` in the previous experiment, even with the model trained on the full PUD corpus. In fact, even though percentages seem similar for English-Swedish, the CSE corpus is roughly half the size of the full PUD corpus.

Results are less encouraging in terms of recall: the number of alignments extracted from the course plans corpora is similar to that obtained from the PUD treebank sample, despite the difference in size. This is explained in part by the presence, in the course plans corpora, of a large amount of very short sentences, and in part by the fact that parse errors introduced by UDPipe make it impossible to align many subsentences without a significant loss in terms of precision.

3.3 MT experiments

The second set of experiments has the objective of assessing the quality of the final output of the system we propose: GF translation lexica. Because we are now focussing on using CA in the context of domain-specific MT, we do not make use of the PUD treebanks, where sentences come from a variety of different sources, but just of the course plans corpora. We do not construct a grammar specific to such domain: for small-scale MT experiments, it is sufficient to extend the extraction grammar itself with preexisting syntax rules defined in the RGL.

The idea is to automatically translate a set of English sentences to Italian and Swedish, ask native speakers of the target languages to produce a set of reference translations, and compare them to the original machine-generated ones by computing BLEU scores.

Due to the small size of the datasets and the consequent low coverage of the extracted lexicon, we generate the sentences to translate directly in the GF shell rather than trying to parse arbitrary sentences other course plans. In order to do that, we make use of GF's random AST generation functionality but at the same time manually select semantically plausible sentences to facilitate the task of the human translators. The results of this process are two small testing corpora, one for the DMI and one for the CSE corpus, each consisting of 50 English sentences.

Reference translations are obtained by asking participants to compare the original English sentences to their automatically translated counterparts and correct the latter by making the minimal changes necessary to obtain a set of grammatically and semantically correct translations. This is important as, if the reference translations are obtained independently from the automatic ones, BLEU scores can easily become misleading.

3.3.1 Results

Corpus-level BLEU scores for the automatic translations of the 50+50 sentences of the testing corpora are summarized in Table 3. Following conventions, we report the cumulative n-gram scores for values of n from 1 to 4 (BLEU-1 to BLEU-4). However, being a significant portion of the sentences of length 4 or less, we also report BLEU-1 to BLEU-3 scores, BLEU-1 to BLEU-2 scores and scores obtained considering unigrams only.

These synthetic figures are useful to give an idea of the general quality of the translations: overall, although with relatively low scores, English-to-Swedish translation works significantly better than

	DMI (en-it)	CSE (en-sv)
BLEU-1 to 4	55	61
BLEU-1 to 3	63	68
BLEU-1 to 2	70	74
BLEU-1	79	81

Table 3: BLEU scores for automatic translations based on the course plans grammars.

	DMI (en-it)	CSE (en-sv)
semantical	23 (46%)	23 (46%)
grammatical	10 (20%)	3 (6%)
both	3 (6%)	4 (8%)

Table 4: Types of errors encountered in the automatically translated sentences.

English-to-Italian. Looking back at the results reported in Section 3.2.2, the reason for this is not immediately clear, as the difference in precision between the two language pairs is negligible on the course plans corpora.

Looking at sentence-level scores can, however, be more insightful. For both corpora, scores assigned to individual segments range from the minimum possible value of 0 to the perfect score of 100, which indicates a perfect correspondence between the automatic and the reference translation.

Examples of sentences that were assigned a perfect BLEU-1 to 4 score are *"the library provides useful textbooks"* (translated to Italian as *"la biblioteca fornisce libri utili"*) in the DMI corpus and *"this lab is more difficult than the exam"* (whose Swedish translation is *"den här laborationen är svårare än tentamen"*) in the CSE corpus.

On the other hand, it is easy for shorter sentences to be assigned the minimum BLEU-1 to 4 score even when they only contain a single grammatical or semantic error. This is the case, for instance, of the sentence *"the test is oral"*, whose last word, *"oral"* is translated as *"dura"* (*"hard"*) instead of *"orale"* due to an alignment error. Interestingly, this is one of many cases in which the correct alignment ⟨*"oral"*,*"orale"*⟩ is also identified by the CE module, but discarded as we only take the first translation candidate into account.

Furthermore, a problem with using the BLEU score as the only evaluation metric is the fact that it makes no distinction between content and function words, thus not allowing an evaluation focused specifically on the extracted concepts. The small size of the corpus, however, allows for some error analysis. From the participants's observations about the kind of errors encountered when manually editing the automatic translation, summarizes in Table 4, we can conclude that while most errors are in fact due to wrong alignments, the main difference between two corpora lies in the number of translations that only contain grammatical errors. This explains the significant difference observed in the cumulative BLEU scores shown in Table 3.

Among other things, Italian contractions are often handled incorrectly. For example, *"del"* (*"di"* + *"il"*, in English *"of the"*) is systematically rendered as two separate words due to UDPipe tokenization. Grammatical errors in Swedish are less common and less systematic. Only in one case, for instance, gender is incorrect (*"programbiblioteken"*). These errors are easy to handle when writing a domain-specific grammar or, in cases like the latter, by making small adjustments to the morphological dictionaries.

Some errors regarding the extracted concepts are also interesting to analyze: the alignment ⟨*"class"*, *"classe"*⟩, for instance, causes the sentence *"I will attend the class"* to be (incorrectly) translated as *"io seguirò la classe"* instead of *"io seguirò la lezione"* even though the correspondence is in fact valid in most of the numerous contexts in which (within the same domain!) *"class"* is not to be intended as a synonym of *"lesson"*.

4 Conclusions

We presented a syntax-based alignment method with a focus on its applications in domain-specific translation lexica generation.

Compared to the existing statistical tools, our system has the following advantages:

- it performs consistently well even on extremely small parallel corpora

- it is able to simultaneously extract correspondences between individual words, multiword expressions and full phrases, even when their constituents are discontinuous

- in conjunction with `gf-ud` pattern matching, it can be used to extract specific types of correspondences, such as predication patterns

- it can automatically generate compilable, morphology-aware GF translation lexica

- it can be configured to easily handle systematic, possibly language-pair or corpus specific translation divergences.

The tangible fruits of this work are a Haskell library and a number of executables offering a user friendly interface to perform CE, lexicon generation and various kinds of evaluations. The source code, including a preliminary implementation of CP, is available at `github.com/harisont/concept-alignment`.

4.1 Current and future work

Our results, while encouraging, suggest that there is room for improvement in many different directions.

An obvious possible development is to improve the current implementation of CP, optimizing it for its different use cases: propagating alignments to a new language looking for correspondences using a translation of the same text they were extracted from or using a different text in the same domain. An alternative to the latter is to make CE, now working on bilingual texts, n-lingual.

When large amounts of data ar available, using our system in conjunction with a statistical tool seems promising. As discussed in Section 2.1.1, this is already partially supported and it could prove useful to develop CA as an actual hybrid system.

Finally, since the freedom that generally characterizes human translation and the quality of currently available UD parsers make maximizing both alignment precision and recall unrealistic, tools to make it easier to postprocess the automatically generated lexica are under development.

References

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Bonnie J. Dorr. 1994. Machine translation divergences: A formal description and proposed solution. *Computational Linguistics*, 20(4):597–633.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia. Association for Computational Linguistics.

Eriksson, Gabrielsson, Hedgreen, Klingberg, Vestlund, and Ödin. 2020. Grammar-based translation of computer science and engineering terminology.

Prasanth Kolachina and Aarnte Ranta. 2016. From abstract syntax to Universal Dependencies. In *Linguistic Issues in Language Technology, Volume 13, 2016*. CSLI Publications.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Hong Kong. Association for Computational Linguistics.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2089–2096, Istanbul, Turkey. European Language Resources Association (ELRA).

Alexandre Rademaker and Francis Tyers, editors. 2019. *Proceedings of the Third Workshop on Universal Dependencies (UDW, SyntaxFest 2019)*. Association for Computational Linguistics, Paris, France.

Aarne Ranta, Krasimir Angelov, Normunds Gruzitis, and Prasanth Kolachina. 2020. Abstract syntax as interlingua: Scaling up the grammatical framework from controlled languages to robust pipelines. *Computational Linguistics*, 46(2):425–486.

Aarne Ranta and Prasanth Kolachina. 2017. From Universal Dependencies to abstract syntax. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 107–116, Gothenburg, Sweden. Association for Computational Linguistics.

Milan Straka, Jan Hajič, and Jana Straková. 2016. UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4290–4297, Portorož, Slovenia. European Language Resources Association (ELRA).

Jörg Tiedemann. 2011. Bitext alignment. *Synthesis Lectures on Human Language Technologies*, 4(2):1–165.