Grammar-Based Concept Alignment for Domain-Specific Machine Translation

Arianna Masciolini
Digital Grammars
arianna@digitalgrammars.com

Aarne Ranta
University of Gothenburg,
Department of Computer Science
and Engineering;
Digital Grammars
aarne.ranta@cse.gu.se

Abstract

Grammar-based domain-specific MT systems are a common use case for CNLs. High-quality translation lexica are a crucial part of such systems, but involve time consuming work and 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 do not always perform well with 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 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 of correctness.

However, lexical exactness is, in this context, just as important as grammaticality. An important part of the design of a Controlled Natural Language (CNL) is the creation of a high-quality translation lexicon, preserving both semantics and grammatical correctness. A translation lexicon is often built manually, which is a time consuming task requiring significant linguistic knowledge. When the task is based on a corpus of parallel example sentences, part of this process can be automated 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 small amount of example data is available — typically, with just one occurrence of each relevant lexical item.

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). A *concept* is a semantic unit expressed by a word or a construction, which is also a unit of *compositional translation*, where translation is performed by mapping concepts to concepts in a shared syntactic structure.

Conceiving concepts as *lemmas* equipped with morphological variations rather than fixed word forms or phrases allows us to generate translation lexica complete with grammatical category and inflection, so that correct target language forms can be selected in each syntactic context.

This paper is structured as follows. Section 2 starts by giving an overview of our approach to CA and comparing it with related work, followed by a description of our CA algorithm. Section 3 presents the results obtained in an evaluation of the system. 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 is analogous to the well-known earlier 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, in this paper we focus on its most mature portion: CE. As results analogous to those that could be obtained via multilingual extraction can be obtained more easily with a combination of CE and CP, we for the moment restrict ourselves to bilingual corpora.

As stated in the Introduction, most existing alignment 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 of different languages and from the lack of robustness of 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, 2011) and Universal Dependencies (UD) (Rademaker and Tyers, 2019).

GF is a constituency grammar formalism 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 depends on, 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, being dependency trees 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 the alignment module. Its function is to extract alignments from parallel bilingual UD treebanks. The outline of the algorithm is given in the following pseudocode:

---

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, and a pair (t, u) of UD trees to align. An example alignment criterion is sameness of syntactic label, for instance, that subjects are aligned with subjects and objects with objects; the details will be discussed in Section 2.1.1. From an imple-

mentation point of view, UD trees are rose trees (trees with arbitrary numbers of branches) where each node represents a word with its dependents as subtrees (see Figure 1). The rose tree is easily obtained from the CoNLL-U notation that UD parsers produce.

As a first step, the program checks whether the two full sentence trees can be aligned with each other, i.e. if they match one or more alignment criteria. In the case of the example criterion discussed above, this means that their roots are labelled the same. If this is the case, they are added to a collection of alignments, which are represented as pairs of UD (sub)trees associated with some metadata, such as the id of the sentence they were extracted from. Such a 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 lack of matching criteria. Subtrees are sorted based on their dependency label to give higher priority to pairs whose heads have the same label (cf. SORT in the pseudocode).

A simple but useful refinement is that, depending on which alignment criteria a pair of trees matches, the heads of the two trees may or may not also be added to the collection of alignments. This is done in order 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 implementation 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 parts of it.

When working on multiple sentences, the algorithm can be applied in an iterative fashion, so that knowledge gathered when a sentence pair is aligned can be used when working on later sentences and to keep track of the number of occurrences of each alignment throughout the entire text. Furthermore, it is possible to initialize the algorithm with a nonempty set of alignments, obtained with the same program or by means of a statistical tool outputting alignments in Pharaoh format and to combine the results of several extraction processes into a single translation lexicon.

2.1.1 Alignment criteria

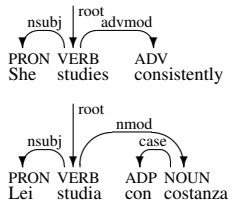
While the alignment criteria are customizable, to allow for a better understanding of the extraction algorithm described above, we explain the criteria that our implementation utilizes by default.

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 a UD tree is always attached to a a fake node with an arc labelled `root`, this criterion also implies that full sentences are always considered to align with each other. This is desirable since we assume that the parallel texts that are fed to our program are sentence-aligned.

Part-Of-Speech equivalence As noted above, the CoNNL-U notation provides 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 function words such as prepositions, determiners and auxiliary verbs, 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_1, t_2 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

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




```
1 She PRON 2 nsubj
2 studies VERB 0 root
3 consistently ADV 2 advmod

1 Lei PRON 2 nsubj
2 studia VERB 0 root
3 con ADP 4 case
4 costanza NOUN 2 obl
```

```
2 studies VERB 0 root
  3 consistently ADV 2 advmod
  1 She PRON 2 nsubj

2 studia VERB 0 root
  1 Lei PRON 2 nsubj
  4 costanza NOUN 2 obl
    3 con ADP 4 case
```

Figure 1: The graphical, simplified CoNNL-U and sorted rose tree representation of a pair of UD sentences. With the default criteria, which among other things allow for matching adverbial with adjectival modifiers, the resulting alignments are: ⟨*She studies consistently, Lei studia con costanza*⟩ (matching `root` label), ⟨*studies, studia*⟩ (head alignment), ⟨*she, lei*⟩ (matching `nsubj` label) and ⟨*consistently, con costanza*⟩ (translation divergence; `amod` and `advmod` treated as equivalent).

grammatical distinctions. When this is the case, it is often straightforward to define alignment criteria 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 this is that of adverbial modifiers (with the UD label `advmod`) translated as prepositional phrases (with the UD label `obl`, for *oblique*), such as in the English-Italian pair ⟨*She studies* ***consistently****, Lei studia* ***con costanza***⟩ (see Figure 1). *Structural divergences*, where a direct object in one language is rendered as an oblique in the other, 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 look like a criterion with no practical applications. However, it can be 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 with a statistical tool and give more credit to correspondences identified by both.

2.1.2 Pattern matching

So far, we have 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 to prioritize precision or coverage, but we did not restrict our search to a particular type of alignments. However, there are cases in which only certain syntactic structures are of interest: for instance, we might be looking for adverbs or noun phrases exclusively.

To handle such cases, the CE module can filter the results based on a `gf-ud` *tree 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 correspondences that cannot be identified by CE alone.

For example, pattern matching can extract verb phrases by looking for full clauses and dropping the subtrees corresponding to subjects. By means of replacements, one can obtain *predication patterns*, that is, 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 alignments outputted by the CE module described so far are represented as pairs of UD (sub)trees in CoNLL-U format. While converting

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

	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.

them to GF ASTs is one of `gf-ud`'s core functionalities, such trees also need be converted into the grammar rules of a compilable GF translation lexicon. To this end, our grammar generation module requires a *morphological dictionary* of the languages at hand and an *extraction grammar*.

Morphological dictionaries, implemented for several languages as part of the RGL, are large collections of lemmas associated with their inflectional forms.

An extraction grammar, on the other hand, defines the syntactic categories and functions the entries of the automatically generated lexicon are built with. For example, entries can be prepositional phrases (PP) or verb phrases (VP) constructed by the following GF functions:

```
PrepNP : Prep -> NP -> PP # case head
PrepPP : VP   -> PP -> VP # head obl
```

The dependency labels appended to the function types instruct `gf-ud` to build GF trees from UD trees that match these labels.

The final translation lexicon, i.e. a GF grammar that extends the extraction grammar, is then derived from these GF trees. In its abstract syntax, the name of each concept is associated with its grammatical category, i.e. the category of the root of the GF tree. For example:

```
fun in_the_field__inom_område_PP : PP ;
```

The concrete syntaxes, on the other hand, contain the linearization rules for each concept, directly based on the trees obtained from `gf-ud`. For instance, in English:

```
lin in_the_field__inom_område_PP =
  PrepNP
    in_Prep
    (DetCN the_Det (UseN field_N)) ;
```

Most function words, such as `in_Prep`, and many content words, such as `field_N`, are available through the morphological dictionaries. When this is not the case, they are assumed to be regular and an additional rule is generated for them. For instance, if the English morphological dictionary didn't contain the word "*field*", we would have:

```
oper field_N = mkN "field" ;
```

3 Evaluation

In this section, we evaluate the system proposed above. We first discuss the data used in the 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 treebanks are available in, we selected English, Italian and Swedish. Using this data limits the amount of alignment errors that are due to annotation issues to a minimum, even though a small number of inconsistencies 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].

3.2 Evaluating CE

While we focus mostly on the MT applications of CA, automatic translation, and much less GF-based domain-specific translation, is not the only context in which CA can be put to use. For instance, it is easy to imagine using it to build translation memories used by human translators. For this reason, a first set of experiments is aimed at evaluating the alignments obtained with our CA module independently from the other 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 statistical tool, `fast_align` (Dyer et al., 2013). In addition, we try to quantify the impact of automated UD parsing 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 forces us to assess the correctness of alignments manually before calculating the ratio between correctly extracted alignments and the total of extracted alignments. What is more, 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 both languages, the idea of missing a chance is expressed with idiomatic expressions very similar to each other. However, the Italian translation mentions a train ("*treno*") in the place of a boat.

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 except with regards to tokenization. On the other hand, the relatively large size of the PUD treebanks makes it possible also to 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 only to extract one-to-many and many-to-one word alignments, given that `fast_align` does not align larger phrases. This explains CE's seemingly low recall. To get a better idea, Table 1 can be compared with Table 2, which summarizes the results of an experiment where no constraints are placed on the size of the extracted alignments.

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, 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 challenges being the inexactness of many translations (which is the direct cause of some of the alignment errors encountered in our evaluation) and the fact that our CE module relies, in this case, on the quality of automatic lemmatization, POS-tagging and parsing. In Table 2, we compare the results obtained on manually annotated data and the course plans corpora parsed with UDPipe.

What is immediately evident, but not unexpected, 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.

The results are less encouraging in terms of recall: the number of alignments extracted from the course plan 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] The pretrained UDPipe models and information about their performance are available at `ufal.mff.cuni.cz/udpipe/1/models`

	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 between the grammar-based extraction of alignments of any size from manually annotated PUD treebanks and from automatically parsed sentences from the course plans corpora.

Our system is, on the other hand, capable of extracting multiword alignments that are unlikely to be identified by a statistical tool, especially in the case of such a small dataset. Examples of this are the noun phrases ⟨*the aim of the course, l'obiettivo del corso, syftet med kursen*⟩ (a concept found in both corpora and, as such, trilingual), ⟨*Natural Language Processing, språkteknologi*⟩ and ⟨*object oriented programming, programmazione ad oggetti*⟩.

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 focusing 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 consequently low coverage of the extracted lexicon, we generate the sentences to translate directly in the GF shell rather than trying to parse arbitrary sentences from 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 with 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, 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.

	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.

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 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 in the course plan 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.

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' 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.

	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.

Among other things, many Italian contractions such as "*del*" ("*di*" + "*il*", in English "*of the*") are 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 ("*programbibliotek**en***"). 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 contexts in which (within the same domain!) *"class"* is not to be intended as a synonym of *"lesson"* but as teaching group.

4 Conclusions

We have presented a syntax-based alignment method with a focus on its applications in domain-specific translation lexicon generation. Compared with the existing statistical tools, our system has the following advantages:

- it performs consistently well even on small parallel corpora
- it is able to simultaneously extract correspondences between individual words, multiword expressions and longer phrases, including discontinuous constituents
- 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.

While they do require manual corrections and completions to an extent that varies according to the quality of the data and the strictness of the criteria used, alignments obtained with our method can reduce the time required for bootstrapping the translation lexicon building process for a domain-specific CNL significantly. In fact, especially if a comprehensive morphological dictionary is available, part of the alignments will be ready to use in a GF-based system without any intervention.

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 on GitHub[4]. The software has already been used to analyse customer-provided data in two commercial projects at Digital Grammars.

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 optimizing the current implementation of concept propagation (CP) for its two 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 former is to make CE, now working on bilingual texts, n-lingual.

When large enough amounts of data are available, using our system in conjunction with a statistical tool seems promising. As discussed above, this

[4] github.com/harisont/concept-alignment

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. 2011. *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Publications, Stanford.

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.