Quantifying training challenges of dependency parsers

Lauriane Aufrant, Guillaume Wisniewski, François Yvon

Quantifying training challenges of dependency parsers

Lauriane Aufrant[1,2], **Guillaume Wisniewski**[1] and **François Yvon**[1]

[1]LIMSI, CNRS, Univ. Paris-Sud, Université Paris-Saclay, 91 405 Orsay, France
[2]DGA, 60 boulevard du Général Martial Valin, 75 509 Paris, France
{lauriane.aufrant,guillaume.wisniewski,francois.yvon}@limsi.fr

Abstract

Not all dependencies are equal when training a dependency parser: some are straightforward enough to be learned with only a sample of data, others embed more complexity. This work introduces a series of metrics to quantify those differences, and thereby to expose the shortcomings of various parsing algorithms and strategies. Apart from a more thorough comparison of parsing systems, these new tools also prove useful for characterizing the information conveyed by cross-lingual parsers, in a quantitative but still interpretable way.

1 Introduction

10 annotated sentences suffice to train a dependency parser that can correctly predict around 70% of the dependencies in French and Italian, but also in Japanese, Greek, Urdu and other languages. This surprising fact raises questions on how to interpret such numbers: what do those 70% dependencies look like? Are they evenly distributed or do they correspond to specific dependency structures?

This work is based on the intuition that in any treebank, some dependencies are intuitively easy to learn (typically nearly deterministic attachments like the dependencies between a noun and its determiner), while others seem much harder to predict, corresponding to complex semantic attachments or long lists of exceptions in the underlying grammar of the language. We thus propose one way to investigate that distinction, by evaluating the amount of training material needed to learn each kind of dependencies: with that viewpoint, a dependency structure is qualified as 'easy to learn' when it can be learned from only a few examples and as 'difficult to learn' otherwise. In this paper, we formalize this idea and propose empirical ways to measure the difficulty to learn certain dependencies.

Our metric proposal can be useful as an analysis tool for comparing existing parsers, which we illustrate both in monolingual and cross-lingual settings, but also to gain some insight on the information reliably conveyed by parsers trained on a few sentences. Ulinski et al. (2016) have indeed shown that such parsers can be successfully leveraged to speed up the manual annotation process. This observation confirms that they provide *some* useful content to the annotators, which our formalism can help characterizing beforehand.

The rest of this paper is organized as follows. Section 2 presents the dependency parsing task and the parsers used in this study. Our method for estimating difficulty is introduced in Section 3. We then apply these newly designed tools to conduct two analyses: a detailed comparison of several state-of-the-art parsers (Section 4), as well as a fine-grained evaluation of cross-lingual transfer (Section 5).

2 Dependency parsing

The purpose of the dependency parsing task is to predict, for each token in a sentence (the child), the token it depends on (its head), and thereby to build a tree structure over the sentence. There are several approaches to build such parsers, mostly corresponding to two categories: transition-based parsers

Proceedings of the 27th International Conference on Computational Linguistics, pages 3191–3202
Santa Fe, New Mexico, USA, August 20-26, 2018.

(which score and apply locally a sequence of transitions affecting the parser's inner state) and graph-based parsers (which compute attachment scores for all pairs of tokens and then optimize the sentence score globally).

In this work, we consider three dependency parsers, based on diverse parsing algorithms and classifiers, to assess the generality of our findings: UDPIPE (transition-based parser, with a feedforward neural classifier, i.e. embedding-based), BEAM (transition-based parser, with an averaged perceptron classifier, i.e. feature-based), and MSTPARSER (graph-based parser). We use version 0.5.1 of MSTPARSER (McDonald et al., 2005) with default parameters. For UDPIPE, we use version 1.1 (Straka and Straková, 2017) with the same hyperparameters as Straka (2017), but without the word embeddings pre-trained on massive monolingual data (to ensure comparability). For BEAM, we rely on our own open source[1] implementation, PANPARSER (Aufrant and Wisniewski, 2016), using the ArcEager version (Nivre, 2004) with a dynamic oracle (Goldberg and Nivre, 2012) adapted for beam search (Aufrant et al., 2017) and the feature sets of Zhang and Nivre (2011) (coarse PoS, no labels).

3 Measuring difficulty

The purpose of this work is to investigate dependencies for the learning of which large training datasets are unnecessary, and which can therefore be qualified as 'easy to learn'. In this section, we formalize this intuition and introduce several empirical measures to quantify it. We then exploit the results of large-scale evaluations to design a new metric, COMPLEXITY, which estimates the challenges faced when learning a given 'type' of dependencies: by departing from individual dependencies, we aim at discovering higher-level properties related to language-independent syntactic phenomena (like the simplicity of annotating determiners, independently of the language and the parsing system).[2]

As our focus is on how parsers exploit the first examples of a training set, even just 10 sentences, we consider in this work only treebanks under 500 sentences.[3]

For all experiments, we use the Universal Dependencies 2.0 dataset (Nivre et al., 2017b; Nivre et al., 2017a), containing 73 treebanks covering a wide array of language families. We first downsize each treebank, when possible, to the 500 first training sentences,[4] and resample smaller trainsets of increasing sizes.[5] Since training is significantly unstable at that scale, all reported scores are averaged over five random samplings, thereby amounting to 5,880 parsers trained on 56 languages. UAS evaluation is performed using gold PoS tags,[6] excluding punctuation.

3.1 Class-level learning rate

Dependency *classes* refer, in the following, to any criterion describing a subset of dependencies (and by extension, a subset of child tokens) in a treebank. The definition of a class can, for instance, be based on child-head distance, in-tree depth, edge direction, dependency label, empirical values like frequency, PoS tags, etc. Even though our method applies to any such criterion, we focus in this section on classes defined by the child PoS and its attachment direction. Indeed, it fits particularly well our intuition regarding many parsing difficulties: due to its frequency and determinism, the ADJ$\overset{\curvearrowleft}{}$ class (that is, all adjectives whose head is on the right) appears for instance simple in the English UD as it mostly corresponds to the bigram 'ADJ NOUN' and, sometimes, to predicative adjectives. On the contrary, the

[1] Source code available at `https://perso.limsi.fr/aufrant`.

[2] Such properties can still depend on the annotation scheme though, as repeatedly pointed out in the literature (Schwartz et al., 2012; Wisniewski and Lacroix, 2017; Wisniewski et al., 2018).

[3] This size has been chosen to cover both the scale of 10 sentences and that of existing treebanks, around 300 sentences: there have been publications with 300 Irish sentences (Lynn et al., 2012) or 371 in Slavomír Čéplö's Maltese treebank (Tiedemann and van der Plas, 2016).

[4] We similarly downsize the validation sets, used only for early stopping, to their first 10 sentences. We do not alter the test sets. We only experiment on the 56 treebanks that are large enough to apply these sampling procedures. `ar_nyuad`, whose complete data is under license, is also excluded.

[5] Resulting trainsets contain 5, 10, 20, 50, 100, 200 and 500 sentences.

[6] While less representative of real-world processing capacities (Tiedemann, 2015), we believe this choice to be crucial in such studies focusing on syntactic properties, whose measurement would otherwise be biased by properties of the taggers and language-dependent vocabulary issues.

attachment decisions on $\overset{\curvearrowright}{\text{ADJ}}$ tokens seem more complex, first of all because the PoS of the head is uncertain (sometimes a VERB, a NOUN, another ADJ, etc.). What remains to ascertain is whether this simple/complex distinction can relate to measurable properties.

Our first experiment aims at studying the rate at which the different dependency classes are learned. In this experiment, we are not interested in the absolute scores over each class, but rather in how fast the *available* information (regarding a given class) can be extracted from a treebank: has everything been already learned in the 10 first sentences? Would 100 more sentences be really informative? Consequently, all measures are performed in terms of UAS normalized by its maximum, in order to set aside the differences due only to the inherent accuracy of a class. In other terms, for class C (e.g. $\overset{\curvearrowright}{\text{DET}}$) and treebank size s (e.g. 50 sentences), we measure $\text{UAS}_s[C]$, the number of tokens in class C whose head has been correctly predicted based on a treebank of size s, and then compute $\frac{\text{UAS}_s[C]}{\text{UAS}_{500}[C]}$. These values are computed separately for each language, and then averaged over all languages[7] to expose language-independent trends, i.e. searching for properties of e.g. determiners as a universal category, while ignoring their idiosyncratic uses in individual languages (e.g. their scarcity in Latin, the annotation scheme for French multi-word expressions containing determiners, etc.).

Figure 1 presents the resulting learning curves for the UAS broken down by class, as well as for the overall score in each language (using BEAM parsers). For legibility (notably around 10 sentences), but also to consider equally the learning speed at all size scales, it is displayed with a logarithmic scale. In this view, the slope of the curve can be interpreted as the marginal utility of doubling the treebank size.

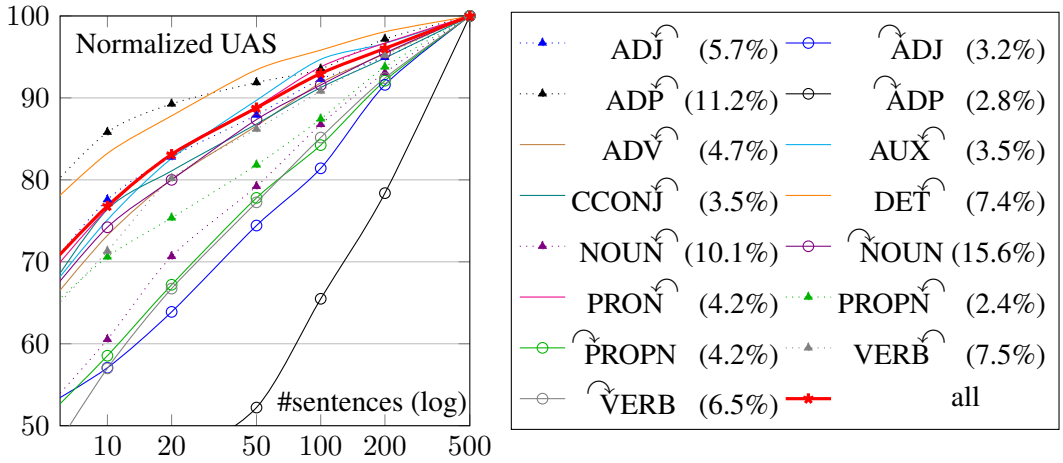

Figure 1: Learning curves of UAS by PoS/direction class. $\overset{\curvearrowright}{\text{P}}$ (resp. $\overset{\curvearrowleft}{\text{P}}$) means all dependencies whose child PoS is P and reference head is on the left (resp. right, including roots). Percentages indicate the size of each class; the rarest ones are not represented.

Unsurprisingly, all fine-grained curves and the overall one are roughly linear: even though they rise quickly to high accuracies, subsequent examples tend to contribute less and less to accuracy, so that the dataset must be doubled each time to achieve substantial UAS improvements. This supports the idea that, overall but also *for each class*, most knowledge is extracted from the first few examples, and additional data is poorly informative.

What is more intriguing in Figure 1 is that the relative curves can clearly be partitioned into two groups,[8] quickly-converging curves and slower-increasing ones, separated by a significant margin. This observation is consistent with our intuition whereby there are two kinds of dependency classes, the simple and the complex ones: as expected, $\overset{\curvearrowleft}{\text{DET}}$ and $\overset{\curvearrowright}{\text{AUX}}$ for instance belong to the former, i.e. the classes for which most of the available information can already be found in the first examples.

[7] To ensure reliable scores, we exclude the treebanks whose test set contains less than 30 occurrences of the considered class.

[8] Because Figure 1 has 3 outliers, it contains in fact 4 groups, but such detailed distinctions are out of the present scope, and the two retained groups are the curves which start at 70% of their final value, and those which start around 50%.

One explanation to these differences in learning rate could be the frequency of classes: those which occur more often are more likely to appear a lot in the first examples, which makes them mechanically easier to learn. However, the ranking of the curves does not seem fully consistent with the size of the corresponding classes: DET has a steeper curve than ADJ, which is itself steeper than NOUN, although ADJ is the least frequent of all three.

Evidence rather suggests that this ranking relates to linguistic properties (like word order entropy), complemented by phenomena of the 'rule versus exception' kind: among competing classes (with different directions for the same PoS, e.g. ADJ versus ADJ), there is systematically one simple and one complex class, in general according to their frequency – but not always, as shown by PROPN. A thorough investigation to explain learning speed differences is however out of the scope of this work, which focuses on measuring them.

3.2 Complexity measures

Although the aforementioned experiment has already revealed interesting trends, a systematic evaluation of complexity requires to go beyond visual reading of curve shapes. But their slope is not properly defined, considering that for several classes, learning speed can show significant variations between 5 and 500 sentences. We consequently aggregate the curve shape into a single metric by computing the area under the curve; yet we calibrate it on the average curve to enable cross-treebank comparison.[9] Formally, we define the COMPLEXITY of a class as the signed area[10] between the learning curve of the (normalized) overall score and that of the class. The logarithmic scale is kept to ensure that the slope at the smallest scale significantly contributes to the metric.[11] A negative COMPLEXITY (curve above average) means that training on this class is simple (most knowledge is embedded in the first few examples), a positive value denotes a complex class (compared to the typical complexity of the treebank). From now on, the terms *simple* and *complex* are respectively used to denote classes with negative and positive COMPLEXITY.

BEAM																	
COMPLEXITY	ADP	DET	PRON	AUX	ADJ	CCONJ	N	ADV	V	PN	SCONJ	N	PN	V	ADJ	AUX	ADP
	-18.8	-18.7	-0.6	0.2	1.9	6.3	7.6	9.6	12.6	23.4	35.0	42.0	49.5	52.5	57.7	68.0	131.2
UAS_{500}	DET	ADP	AUX	PRON	SCONJ	ADJ	CCONJ	ADV	V	PN	PN	N	N	ADJ	V	AUX	ADP
	91.3	89.0	83.9	82.4	80.2	80.0	77.1	76.1	75.1	69.0	68.4	68.2	67.9	60.6	56.4	52.8	48.0
UDPIPE																	
COMPLEXITY	ADP	AUX	CCONJ	DET	ADV	ADJ	PRON	PN	N	V	AUX	N	SCONJ	ADJ	ADP	PN	V
	-33.3	-24.5	-5.3	-3.7	-2.3	0.7	3.3	12.5	25.3	26.0	28.3	35.5	35.9	45.1	51.0	51.8	75.1
UAS_{500}	DET	ADP	AUX	SCONJ	PRON	ADJ	CCONJ	ADV	V	PN	N	N	PN	ADJ	V	AUX	ADP
	89.9	89.0	84.3	81.0	80.3	79.9	76.1	74.6	73.9	73.9	69.9	65.6	65.0	56.9	53.8	44.4	42.2
MSTPARSER																	
COMPLEXITY	ADP	DET	ADJ	V	CCONJ	PRON	PN	ADV	N	AUX	N	ADJ	SCONJ	PN	AUX	V	ADP
	-27.7	-24.1	-15.4	-3.5	6.2	8.7	13.1	13.8	23.4	28.9	34.2	47.4	53.8	55.5	56.7	83.8	89.1
UAS_{500}	DET	ADP	AUX	ADJ	PRON	V	CCONJ	ADV	SCONJ	N	PN	PN	N	ADJ	AUX	V	ADP
	90.6	89.3	80.8	79.8	79.6	76.0	74.9	74.3	72.7	67.8	67.2	67.1	64.7	57.0	56.4	50.6	50.5

Table 1: Comparison of the COMPLEXITY and UAS rankings for all 3 systems. For each system, the classes with the largest difference between both rankings are highlighted in blue. N, PN and V stand for NOUN, PROPN and VERB.

Beyond a sign-based grouping criterion, these quantitative measures also provide an interesting diffi-

[9] This aggregation step is inspired by Zubek and Plewczynski (2016)'s work on data complexity, which is close in spirit to ours (with the same intuition that a dataset is simple if most information can be found in the first few examples) but explores a different setting: in their work, they measure the similarity of subsets of increasing sizes to the dataset itself (without involving a prediction step) and aggregate these values to estimate the complexity of that dataset.

[10] This can be computed easily using any numerical integration method. In the following experiments, we use Simpson's rule.

[11] In these experiments the area is computed between 5 and 500 training sentences, but another low-resource range could be chosen, provided that it is the same for all treebanks.

culty ranking. Table 1 compares the rankings based on UAS (using 500 training sentences) and COMPLEXITY. Overall, the results remain consistent with intuition: the simplest classes are closed PoS classes with very deterministic attachments (DET$^{\curvearrowleft}$, ADP$^{\curvearrowleft}$), the most complex ones are rare ($^{\curvearrowright}$ADP) or semantics-driven attachments ($^{\curvearrowright}$VERB), while classes of average difficulty are mostly NOUNs and other open classes. But differences between both metrics still appear clearly, thereby confirming that they capture different properties. For instance, high accuracies are achieved on SCONJ$^{\curvearrowleft}$, but getting there is cumbersome as it requires many training examples. Conversely, UDPIPE does not learn much about the $^{\curvearrowright}$AUX class (whose UAS is particularly low), but it does so almost immediately (as shown by its low complexity).

Finally, we perform similar computations on each treebank separately:[12] they reveal that, even though some classes have emerged as consistently simple across languages, this property can still depend on the language (sometimes even on the corpus). For instance, ADJ$^{\curvearrowleft}$ is simple and $^{\curvearrowright}$ADJ is complex in the English treebank, but in French it is the opposite, which is consistent with the respective frequencies of those classes (and thus with the initial intuition). Similarly, $^{\curvearrowright}$ADP is complex for all but 6 treebanks (from which the competing class ADP$^{\curvearrowleft}$ is virtually absent), and DET$^{\curvearrowleft}$, whose attachments are highly deterministic, is a simple class for all but 7 treebanks (mostly Old Church Slavonic, Arabic, Latin-PROIEL, but also Basque, Estonian, Korean and Polish to a lesser extent).

4 Application 1: fine-grained comparison of parsers

The metrics we have proposed can now be used for large-scale computation of fine-grained evaluations, and therefore detailed comparison of parsers. This newly defined notion of complexity opens indeed new evaluation perspectives, as a complement to the more explicit properties (length, PoS tags, projectivity, etc.) used in prior work on comparative error analysis (McDonald and Nivre, 2007).

Complexity variations Coming back to Table 1, comparing the rankings between all 3 systems also emphasizes on their respective shortcomings. UDPIPE notably seems to have troubles with determiners: not only does it achieve a lower score on DET$^{\curvearrowleft}$ dependencies, but it is also much slower learning those, compared to the other systems; UDPIPE consequently appears to under-exploit the determinism of that class.[13] It is conversely particularly efficient on CCONJs. As for MSTPARSER, it handles VERB$^{\curvearrowleft}$ significantly faster than BEAM and UDPIPE, presumably because it does not rely on mostly local features, as transition-based parsers do. Regarding the BEAM system, its main particularity is its efficient handling of AUX$^{\curvearrowleft}$ attachments, even though their UAS is around the same. Overall, according to Figure 2, COMPLEXITY correlates well with UAS_{500}, at least for BEAM and MSTPARSER (Spearman's ρ = -.886 and -.882); but it is hardly the case for UDPIPE (ρ = -.576), which points out at least a difference, and maybe an issue, which remains to ascertain. All those remarks provide a valuable feedback on the inner workings of each parsing system, and thereby new insights on possible issues. The COMPLEXITY metric thus provides a promising way to analyze and improve parsing algorithms in general.

Composite scores A more systematic way to study fine-grained differences between parsers is to investigate composite scores. However, for large inventories of classes, class-level analysis can be tedious and hide the main trends. For this reason, we advocate aggregating that information into 2 scores only, as a trade-off between simplicity and detailed analysis. Contrasting differences among parsers along two categories, simple and complex dependencies in the present case, is indeed already much more informative than relying on a single metric.

Table 2 reports the average UAS of the 3 parsers, when computed separately on simple and complex classes (that is, depending on the sign of COMPLEXITY), and for various data sizes. It appears that all

[12] In other words, we compute the area under a specific learning curve, instead of the area under the average of all learning curves.

[13] As UDPIPE is the only neural parser in our experiments, this differential treatment of deterministic classes may be related to the choice of classifier; concluding on such properties is out of the scope of this paper, though.

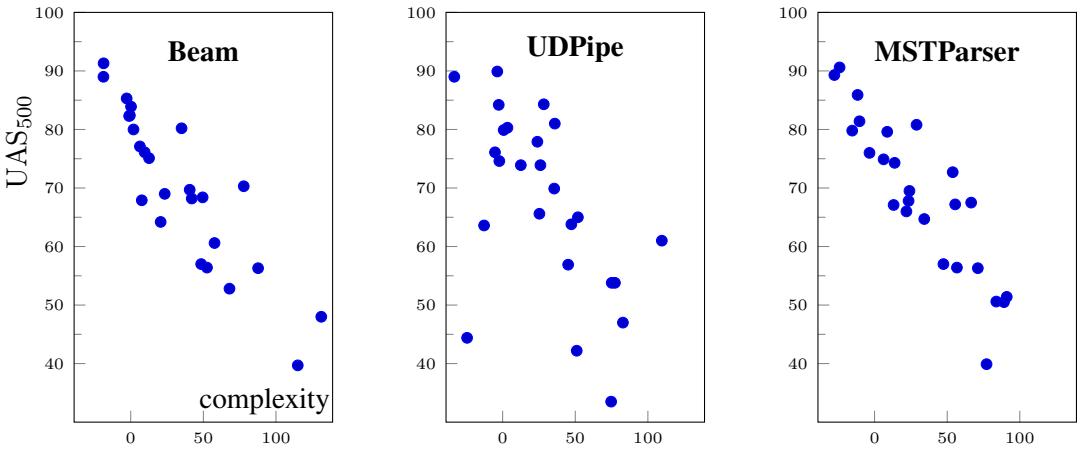

Figure 2: UAS of all 3 systems, with 500 training sentences, for classes of increasing complexity.

systems present a significant score difference (up to 30 points) between both categories, in favor of the simple one, in particular for tiny data (i.e. 10 sentences).

It is important to note that this result was not expected: it is true that UAS measures are involved when computing COMPLEXITY, but all these scores are first normalized. Therefore, the simple classes are not *by design* those with high accuracies on tiny data, despite what the results suggest.[14] In this regard, the score difference between both groups is an actual finding. Besides, the score gap is mostly maintained for full datasets, which means that the classes that have been singled out using tiny data have truly different properties (or at least are handled differently by the systems), whatever the data size.

	UAS_{10}			UAS_{500}			$UAS_{\text{full UD}}$		
	simple	overall	complex	simple	overall	complex	simple	overall	complex
UDPIPE	52.8	42.5	28.4	81.8	74.7	65.2	87.6	83.2	**77.3**
MSTPARSER	64.4	52.8	36.9	82.7	75.1	64.9	**88.2**	**83.4**	77.1
BEAM	**71.1**	**59.0**	**42.7**	**82.9**	**76.1**	**67.1**	87.3	82.6	76.4

Table 2: UAS of the 3 studied systems, broken down by simple and complex PoS/direction classes, and for various data sizes. The partitions and corresponding scores are computed separately for each language, before averaging. All partitions are computed using BEAM, so that the absolute scores remain comparable (best system in bold). 'Full UD' trainsets contain between 598 and 68,495 sentences.

When comparing each system's results, the dynamics between them become clear: while BEAM prevails across the categories around 10 sentences, MSTPARSER catches up first on simple dependencies (for 500 sentences), then also on the complex ones (for full UD). As for UDPIPE, it performs poorly with 10 training sentences, but then gradually reveals its worth for parsing specifically complex classes (which is consistent with its difficulties with determiners: this system is not so relevant for deterministic dependencies, its value is elsewhere).

5 Application 2: fine-grained evaluation of cross-lingual parsers

Apart from comparing systems, our proposal can also be used to compute reference values when evaluating other kinds of parsers, typically cross-lingual parsers. Cross-lingual transfer is an approach to process low-resourced languages, whereby resources available in other languages (the sources) are leveraged to

[14]There remains however an indirect impact of absolute scores on COMPLEXITY: the classes that reach already high scores (over 80 UAS) with only 10 sentences are doomed to be simple classes, because being complex would imply achieving more than 100 UAS with 500 sentences. So, there is indeed a score bias towards simple classes. But such classes with early high scores represent a minority of the dependencies (19% on average), and there are many other kinds of simple classes (for a total of 57% of dependencies), including poorly accurate ones.

build new systems in the language of interest (the target). The main methods involve typically projecting information through word-aligned parallel data (Yarowsky and Ngai, 2001), or training source models on delexicalized data (that is, using PoS information only) and directly applying them on the target side (Zeman and Resnik, 2008). The link with our work above is twofold: cross-lingual parsers also focus on low ranges of training sizes, and at the same time many of them yield UAS around the range covered by our 5 to 500-sentence long treebanks. We therefore propose to exploit our upper results in this new frame, with the goal of quantifying and characterizing the amount of knowledge that has been effectively transferred: what kind of information is learned by cross-lingual parsers – only simple classes or complex knowledge about non-trivial classes?

Multi-source weighted delexicalized transfer Our analysis first focuses on Rosa and Zabokrtsky (2015)'s state-of-the-art method for cross-lingual parsing: it consists in delexicalized transfer, where the hypotheses stemming from multiple sources are weighted and combined based on the KL_{cpos^3} metric (the Kullback-Leibler divergence of PoS trigram distributions between the source and the target). It is meant to favour the languages that are syntactically close to the target, while still benefiting from the diverse information conveyed by a large set of sources.

We reimplement the method of Rosa and Zabokrtsky (2015) on top of the BEAM system: we include as sources the delexicalized BEAM models based on all 56 (full) treebanks except those covering the same language. All scores are again averaged over 5 different trainings. To be fair regarding the available target resources, the KL_{cpos^3} metric is computed using the whole source treebank but only 10 target sentences. In the following, this strategy is referred to as KL-BEAM.

Composite scores Similarly to what has been done for monolingual parsers, we can express the performance of KL-BEAM in terms of simple-only UAS and complex-only UAS (using the partition computed monolingually with BEAM for instance). Those values alone are, however, hard to interpret, and notably to relate to the actual amount of knowledge transferred via KL-BEAM.

We therefore propose to position those scores along the learning curves of the monolingual parser, following Aufrant et al. (2016): if the cross-lingual parser achieves the same score as a parser trained on n sentences, we consider that the amount of transferred knowledge is the amount of knowledge contained in n sentences. Figure 3 consequently pictures the learning curves of each system on simple and complex classes (using the PoS/direction criterion), as well as the split UAS for KL-BEAM on the same categories.

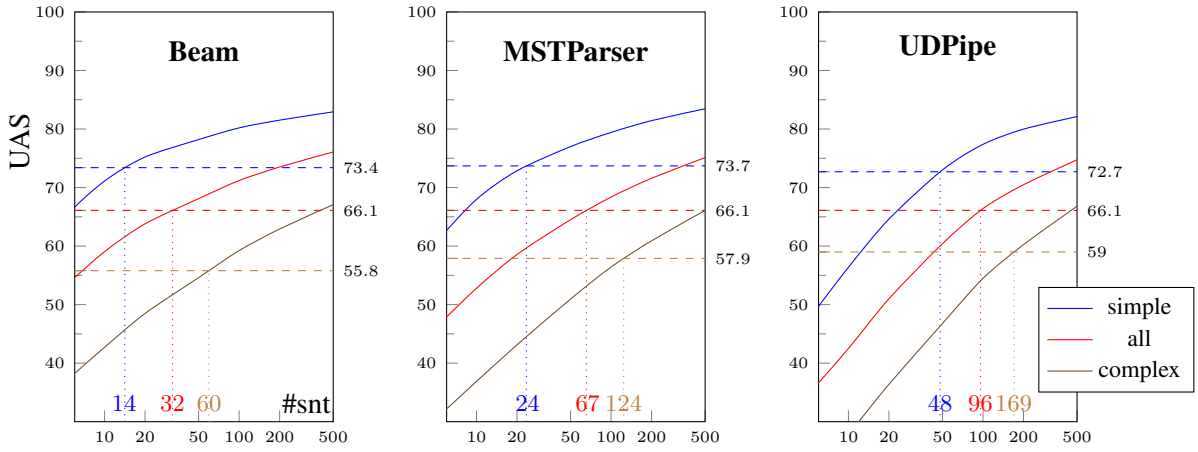

Figure 3: Overall and simple/complex UAS of all 3 parsers with increasing training sizes. The partitions and corresponding scores are computed separately for each language and system, before averaging. Dashed lines correspond to the overall, simple and complex UAS for KL-BEAM, using the partition computed for each system. Dotted lines represent the corresponding numbers of annotated training sentences.

Overall, KL-BEAM achieves 66.1 UAS on average, which corresponds to what could have been achieved by BEAM with 32 sentences. In other words, if more than 32 target sentences are available,

on average one should prefer monolingual parsing (at least when using BEAM) over KL-BEAM transfer:[15] the amount of knowledge transferred across languages seems very limited.

However, when considering the simple/complex scores, it appears that their difference is less pronounced for the cross-lingual parser than the monolingual ones, which results in distinct data-size equivalents. Delexicalized transfer thus appears much more useful for complex classes (score equivalent to 60 sentences) than for simple classes, for which few information was transferred (score equivalent to 14 sentences). This means that KL-BEAM is qualitatively better than a BEAM trained on 32 sentences: it is able to convey non-trivial information, which otherwise would have only been accessible by collecting more data.

As UDPIPE and MSTPARSER are less efficient on very small treebanks, the data-size equivalents are much higher, but the simple/complex difference remains similar in proportions. We additionally perform similar experiments with other class criteria and obtain similar results, although the simple/complex gap marginally fluctuates: for BEAM, we measure 20 versus 43 sentences (74.6 and 58.9 UAS) when considering only the PoS, 15 versus 60 sentences (74.6 and 56.4 UAS) when considering the relation label only, and 13 versus 67 sentences (74.8 and 53.6 UAS) when using the PoS, label and direction at the same time.

Other non-conventional parsers The observations made in this section shed a new light on the performance of state-of-the-art parsers trained using non-conventional data. The accuracies achieved with very small datasets are indeed comparable to other scores reported in the recent literature.

In the original experiments conducted with multi-source weighted delexicalized transfer (not based on BEAM), Rosa and Zabokrtsky (2015) achieve an UAS of 52.5 on HamleDT 2.0; in another line of cross-lingual parsing, the projection technique of Tiedemann and Agić (2016) yields 75.43 UAS on UDT 1.0 (COMBG model). Although neither work is conducted on UD 2.0, and thus the comparison is inexact, our metrics can still provide a rough estimation of each method's success: using UD 2.0, these scores would have been achieved by training on, respectively, 5 and 444[16] sentences. Empirical evidence hence raises serious doubts on the actual effectiveness of the main transfer techniques: cross-lingual parsers embed a limited amount of knowledge, and do not save much annotation effort. This is less true for methods based on projection, but these already require large data and significant human efforts to find, clean and sentence-align parallel data – a work which may turn out to be more demanding than annotating a small treebank.

Unsupervised parsing is another field in which parsers are trained in a non-conventional way, and would thus benefit from a comparison with monolingual supervision. For instance, Mareček and Straka (2013) achieve 48.7 UAS on average over the CoNLL 2006-2007 treebanks, while the average UAS of the rule-based approach of Martínez Alonso et al. (2017) is 57.5 on UD 1.2. These UAS correspond to 4 and 9 annotated sentences, which, again despite the dataset mismatch, still highlights how much work remains to be done in that field.

Size-based evaluation can consequently provide interesting reference points for parsers stemming from many fields: cross-lingual transfer and unsupervised models, but also many others like domain adaptation for instance.

Information conveyed by one source As a final experiment, we propose to evaluate the contribution of a given source to multi-source transfer, in terms of the amount of conveyed information.

[15] Because it is an average, this boundary should not be interpreted in a strict sense but as a trend: it is not applicable to each individual language in the considered set. The ability to transfer syntactic information depends indeed a lot on the linguistic setting (standard deviation of 15.3 UAS for KL-BEAM), so that the averages presented here do not replace any prior knowledge on transfer results for a given known language. However, when tackling a *new* language, they can provide a rough estimate to choose between the monolingual and cross-lingual approaches.

[16] While this amount can appear already substantial, it also denotes the fact that Tiedemann and Agić (2016)'s evaluation is based on a set of languages with marked relatedness, and thus good transfer properties. When averaging our measures only over that set of languages (but still not the same treebanks), the data-size equivalent drops to 92 sentences. Similarly, Lacroix et al. (2016)'s projection approach achieves 79.62 UAS, which is higher than with 500 sentences when considering all UD 2.0 treebanks, but corresponds to 295 sentences when retaining the same set of languages. Overall, the magnitude remains around a few hundred sentences for projection methods.

We build a cross-lingual parser for Romanian, using KL-BEAM as before but with a much smaller source set: French, Italian, Spanish and Bulgarian. The choice of the first three is motivated by their language family (Romance, like Romanian), and Bulgarian by geographic proximity (and hence various influences across history). This parser is then compared (overall and also along simple/complex classes) with those obtained when removing either one of the sources, in order to characterize precisely their contribution.

Table 3 reports the overall and simple/complex UAS measures, as well as data-size equivalents based on the Romanian BEAM. Comparing the full source set with the reduced ones, the respective benefits of each source appear clearly: Italian and French convey mostly complex information (respectively +0.3/+1.8 and +0.2/+1.2 on simple/complex scores), while Bulgarian improves mostly on simple classes (+1.5/+0.2 on simple/complex scores) and Spanish contributes equally in both cases (+1.5/+1.2).

Sources	UAS (ro)			#sentences (ro)		
	simple	overall	complex	simple	overall	complex
fr + it + es + bg	81.4	74.4	60.3	167	213	231
✗ it es bg	81.2	73.8	59.1	149	165	179
fr ✗ es bg	81.1	73.6	58.5	142	155	162
fr it ✗ bg	79.9	73.0	59.1	77	131	179
fr it es ✗	79.9	73.3	60.1	77	142	219

Table 3: Cross-lingual UAS and equivalent monolingual supervision in Romanian, for a KL-BEAM parser trained with various sets of sources. The simple/complex partition is that of BEAM in Romanian.

Regarding Spanish and Bulgarian, it is interesting to note that combining both sources more than doubles the amount of knowledge on simple classes (from 77 to 167). This suggests that multi-source combination itself enables valuable interactions between languages, thereby granting access to knowledge that would have been hard to obtain otherwise.

The results for Bulgarian are worth a closer look: the kind of information it conveys corresponds to syntactic structures that are easy to learn in Romanian (possibly deterministic), and yet not provided by other (Romance) sources. The Bulgarian influence on Romanian – in other words the structures that are out of the scope of Romance syntax – consequently seems to materialize as straightforward dependencies on unambiguous patterns.

On a final note, looking at composite scores instead of UAS only safeguards against overrating a given language: for instance, even though Spanish seems more valuable as a source than Italian (73.6 vs 73.0), it is only so because of the (numerous) simple dependencies, and it actually underperforms Italian on complex dependencies, where the true challenge is though.

6 Conclusion

In this work, we have introduced a new metric, called COMPLEXITY, to evaluate the difficulty to learn a given class of dependencies, in terms of data requirements: inspired by an in-depth analysis of fine-grained learning curves, we have aggregated their information into a single value. A series of systematic computations using that metric has unveiled interesting properties of the 3 considered parsing algorithms; it has notably revealed the kind of dependencies for which the parsers make an abnormally high amount of efforts during training, and hinted at some syntactic properties that they under-exploit.

Computing composite scores by separating simple and complex dependencies has been further enlightening for analyzing the aforementioned parsers, but above all it has enabled an extensive assessment of the benefits of parsers trained in a non-conventional way, typically cross-lingual parsers. While the typical UAS achieved by delexicalized transfer remain low and can easily be obtained with just a sample of target-side supervision, the information conveyed by cross-lingual parsers has in fact appeared of better quality: their actual added value is on complex dependencies. As a final case study, we have

leveraged the notion of complexity to characterize the contribution of a given source to a multi-source transfer procedure.

Regarding the notion of complexity, additional experiments with other class criteria can be found in the PhD thesis of the first author (Aufrant, 2016), as well as some insights on what makes a class complex, and proposals on how to leverage the COMPLEXITY information to improve cross-lingual transfer.

A natural continuation of this work would be to reproduce these measures with other cross-lingual transfer algorithms, and also with other kinds of non-conventional parsers, like unsupervised or out-of-domain parsers. Since the proposed procedure for estimating COMPLEXITY is in fact not specific to dependency parsing (only the class criterion is), it may also be interesting to apply it to other sequence labeling tasks, for instance PoS tagging.

Acknowledgments

This work has been partly funded by the French *Direction générale de l'armement* and by the *Agence Nationale de la Recherche* (ParSiTi project, ANR-16-CE33-0021).

References

Lauriane Aufrant and Guillaume Wisniewski. 2016. PanParser: a Modular Implementation for Efficient Transition-Based Dependency Parsing. Technical report, LIMSI-CNRS, March.

Lauriane Aufrant, Guillaume Wisniewski, and Franois Yvon. 2016. Cross-lingual and Supervised Models for Morphosyntactic Annotation: a Comparison on Romanian. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, 5. European Language Resources Association (ELRA).

Lauriane Aufrant, Guillaume Wisniewski, and François Yvon. 2017. Don't Stop Me Now! Using Global Dynamic Oracles to Correct Training Biases of Transition-Based Dependency Parsers. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 318–323, Valencia, Spain, 4. Association for Computational Linguistics.

Lauriane Aufrant. 2016. *Training parsers for low-resourced languages: improving cross-lingual transfer with monolingual knowledge*. Ph.D. thesis, Université Paris-Saclay.

Yoav Goldberg and Joakim Nivre. 2012. A Dynamic Oracle for Arc-Eager Dependency Parsing. In *Proceedings of COLING 2012*, pages 959–976, Mumbai, India, 12. The COLING 2012 Organizing Committee.

Ophélie Lacroix, Lauriane Aufrant, Guillaume Wisniewski, and François Yvon. 2016. Frustratingly Easy Cross-Lingual Transfer for Transition-Based Dependency Parsing. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1058–1063, San Diego, California, 6. Association for Computational Linguistics.

Teresa Lynn, Ozlem Cetinoglu, Jennifer Foster, Elaine U Dhonnchadha, Mark Dras, and Josef van Genabith. 2012. Irish Treebanking and Parsing: A Preliminary Evaluation. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uur Doan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, 5. European Language Resources Association (ELRA).

David Mareček and Milan Straka. 2013. Stop-probability estimates computed on a large corpus improve Unsupervised Dependency Parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 281–290, Sofia, Bulgaria, 8. Association for Computational Linguistics.

Héctor Martínez Alonso, Željko Agić, Barbara Plank, and Anders Søgaard. 2017. Parsing Universal Dependencies without training. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 230–240, Valencia, Spain, 4. Association for Computational Linguistics.

Ryan McDonald and Joakim Nivre. 2007. Characterizing the Errors of Data-Driven Dependency Parsing Models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131, Prague, Czech Republic, 6. Association for Computational Linguistics.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajic. 2005. Non-Projective Dependency Parsing using Spanning Tree Algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada, 10. Association for Computational Linguistics.

Joakim Nivre, Željko Agić, Lars Ahrenberg, Lene Antonsen, Maria Jesus Aranzabe, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, John Bauer, Kepa Bengoetxea, Riyaz Ahmad Bhat, Eckhard Bick, Cristina Bosco, Gosse Bouma, Sam Bowman, Aljoscha Burchardt, Marie Candito, Gauthier Caron, Gülşen Cebirolu Eryiit, Giuseppe G. A. Celano, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Silvie Cinková, Çar Çöltekin, Miriam Connor, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Marhaba Eli, Ali Elkahky, Tomaž Erjavec, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökrmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta Gonzáles Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Nizar Habash, Jan Hajič, Jan Hajič jr., Linh Hà M, Kim Harris, Dag Haug, Barbora Hladká, Jaroslava Hlaváčová, Petter Hohle, Radu Ion, Elena Irimia, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşkara, Hiroshi Kanayama, Jenna Kanerva, Tolga Kayadelen, Václava Kettnerová, Jesse Kirchner, Natalia Kotsyba, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Tatiana Lando, Phng Lê Hng, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Gustavo Mendonça, Anna Missilä, Verginica Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Shunsuke Mori, Bohdan Moskalevskyi, Kadri Muischnek, Nina Mustafina, Kaili Müürisep, Pinkey Nainwani, Anna Nedoluzhko, Lng Nguyn Th, Huyn Nguyn Th Minh, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Petya Osenova, Lilja Øvrelid, Elena Pascual, Marco Passarotti, Cenel-Augusto Perez, Guy Perrier, Slav Petrov, Jussi Piitulainen, Emily Pitler, Barbara Plank, Martin Popel, Lauma Pretkalniņa, Prokopis Prokopidis, Tiina Puolakainen, Sampo Pyysalo, Alexandre Rademaker, Livy Real, Siva Reddy, Georg Rehm, Larissa Rinaldi, Laura Rituma, Rudolf Rosa, Davide Rovati, Shadi Saleh, Manuela Sanguinetti, Baiba Saulīte, Yanin Sawanakunanon, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Lena Shakurova, Mo Shen, Atsuko Shimada, Muh Shohibussirri, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Antonio Stella, Jana Strnadová, Alane Suhr, Umut Sulubacak, Zsolt Szántó, Dima Taji, Takaaki Tanaka, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Gertjan van Noord, Viktor Varga, Veronika Vincze, Jonathan North Washington, Zhuoran Yu, Zdeněk Žabokrtský, Daniel Zeman, and Hanzhi Zhu. 2017a. Universal Dependencies 2.0 – CoNLL 2017 Shared Task Development and Test Data. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University.

Joakim Nivre, Željko Agić, Lars Ahrenberg, et al. 2017b. Universal Dependencies 2.0. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University, Prague.

Joakim Nivre. 2004. Incrementality in Deterministic Dependency Parsing. In Frank Keller, Stephen Clark, Matthew Crocker, and Mark Steedman, editors, *Proceedings of the ACL Workshop Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57, Barcelona, Spain, 7. Association for Computational Linguistics.

Rudolf Rosa and Zdenek Zabokrtsky. 2015. KLcpos3 - a Language Similarity Measure for Delexicalized Parser Transfer. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 243–249, Beijing, China, 7. Association for Computational Linguistics.

Roy Schwartz, Omri Abend, and Ari Rappoport. 2012. Learnability-Based Syntactic Annotation Design. In *Proceedings of COLING 2012*, pages 2405–2422, Mumbai, India, 12. The COLING 2012 Organizing Committee.

Milan Straka and Jana Straková. 2017. Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada, 8. Association for Computational Linguistics.

Milan Straka. 2017. CoNLL 2017 Shared Task - UDPipe Baseline Models and Supplementary Materials. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University.

Jörg Tiedemann and Željko Agić. 2016. Synthetic Treebanking for Cross-Lingual Dependency Parsing. *Journal of Artificial Intelligence Research*, 55:209–248.

Jörg Tiedemann and Lonneke van der Plas. 2016. *Bootstrapping a dependency parser for Maltese – a real-world test case*.

Jörg Tiedemann. 2015. Cross-Lingual Dependency Parsing with Universal Dependencies and Predicted PoS Labels. In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*, pages 340–349, Uppsala, Sweden, 8. Uppsala University, Uppsala, Sweden.

Morgan Ulinski, Julia Hirschberg, and Owen Rambow. 2016. Incrementally Learning a Dependency Parser to Support Language Documentation in Field Linguistics. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 440–449, Osaka, Japan, 12. The COLING 2016 Organizing Committee.

Guillaume Wisniewski and Ophélie Lacroix. 2017. A Systematic Comparison of Syntactic Representations of Dependency Parsing. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 146–152, Gothenburg, Sweden, 5. Association for Computational Linguistics.

Guillaume Wisniewski, Ophlie Lacroix, and Franois Yvon. 2018. Automatically Selecting the Best Dependency Annotation Design with Dynamic Oracles. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 401–406, New Orleans, Louisiana, 6. Association for Computational Linguistics.

David Yarowsky and Grace Ngai. 2001. Inducing multilingual POS taggers and NP brackets via robust projection across aligned corpora. In *Proceedings of NAACL*.

Daniel Zeman and Philip Resnik. 2008. Cross-Language Parser Adaptation between Related Languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*, pages 35–42.

Yue Zhang and Joakim Nivre. 2011. Transition-based Dependency Parsing with Rich Non-local Features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 188–193, Portland, Oregon, USA, 6. Association for Computational Linguistics.

Julian Zubek and Dariusz M. Plewczynski. 2016. Complexity curve: a graphical measure of data complexity and classifier performance. *PeerJ Computer Science*, 2:e76.