An empirical study of differences between conversion schemes and annotation guidelines*

Anders Søgaard
Center for Language Technology
University of Copenhagen
DK-2300 Copenhagen S
soegaard@hum.ku.dk

Abstract

We establish quantitative methods for comparing and estimating the quality of dependency annotations or conversion schemes. We use generalized tree-edit distance to measure divergence between annotations and propose theoretical learnability, derivational perplexity and downstream performance for evaluation. We present systematic experiments with tree-to-dependency conversions of the Penn-III treebank, as well as observations from experiments using treebanks from multiple languages. Our most important observations are: (a) parser bias makes most parsers insensitive to non-local differences between annotations, but (b) choice of annotation nevertheless has significant impact on most downstream applications, and (c) while learnability does not correlate with downstream performance, learnable annotations will lead to more robust performance across domains.

1 Introduction

Syntactic structures in dependency parsing are moving targets. While intrinsic evaluations often give the impression that syntactic structures are carved in stone, in reality we have little evidence in favor of the structures we posit. While most linguists agree on how to analyze core syntactic phenomena, there is widespread disagreement about a number of cases. Do auxiliary verbs head main verbs? Do prepositions head their nominal complements? And how should we analyze punctuation?

Many dependency treebanks are created by automatic conversion from pre-existing constituency treebanks. Since there exist linguistic phenomena whose analyses linguist do not agree on, it comes as no surprise that different conversion schemes have been proposed over the years (Collins, 1999; Yamada and Matsumoto, 2003; Johansson and Nugues, 2007). The output of these schemes differ considerably in their choices concerning head status and dependency relation inventories (Schwartz et al., 2012; Johansson, 2013).

The number of languages for which we have several dependency treebanks, is limited (Johansson, 2013), but the availability of different tree-to-dependency conversion schemes raises the question of what scheme is better? So does the existence of different parsers relying on different linguistic formalisms, but whose output can be mapped to dependencies (Tsarfaty et al., 2012). But is the question of which is better really a meaningful question? Better at what?

Schwartz et al. (2012) propose to evaluate conversion schemes in terms of learnability. We argue that while learnability is relevant to assess the robustness of dependency schemes, the most important parameter when choosing conversion schemes in practice is down-stream performance. We cite Elming et al. (2013), who show that down-stream performance is very sensitive to choice of conversion scheme. We also suggest that derivational perplexity (Søgaard and Haulrich, 2010) is a less biased measure of robustness than learnability – at least the way it is measured in Schwartz et al. (2012).

The paper presents (a) an empirical analysis of distance between conversion schemes, (b) an

*Section 5 is joint work with Jakob Elming, Anders Johannsen, Sigrid Klerke, Emanuele Lapponi and Hector Martinez, published at NAACL 2013.

Proceedings of the Second International Conference on Dependency Linguistics (DepLing 2013), pages 298–307,
Prague, August 27–30, 2013.

Clear cases		Difficult cases	
Head	Dependent	?	?
Verb	Subject	Auxiliary	Main verb
Verb	Object	Complementizer	Verb
Noun	Attribute	Coordinator	Conjuncts
Verb	Adverbial	Preposition	Nominal
			Punctuation

Figure 1: Clear and difficult cases in dependency annotation.

analysis of the theoretical learnability of conversion schemes, (c) a complexity analysis of conversion schemes in terms of derivational perplexity, and (d) empirical evaluations of the downstream usefulness of conversion schemes. Section 2 introduces a few common conversion schemes and their linguistic differences. In our empirical analyses we will focus on standard conversions of the Wall Street Journal section of the Penn-III treebank of English (Marcus et al., 1993). Section 3 introduces three distance metrics defined over pairs of output dependency structures. The results presented in this section suggests that parser bias cancels out many of the differences between conversion schemes. Section 4 discusses the learnability and derivational perplexity of tree-to-dependency conversion schemes. Section 5 presents a series of experiments, evaluating the downstream performance of conversion schemes in negation scope resolution, sentence compression, statistical machine translation, semantic role labeling and author perspective classification. Section 6 concludes with a discussion of the analyses presented in the previous sections. In our experiments we will use the publicly available MATE parser (Bohnet, 2010). Obviously the downstream performance of a conversion scheme depends on the parsing model chosen and how syntactic features are incorporated in the downstream task, but we do not vary parser or syntactic feature representations in our experiments.

2 Tree-to-dependency conversion schemes

Annotation guidelines used in modern dependency treebanks and tree-to-dependency conversion schemes for converting constituent-based treebanks into dependency treebanks are typically based on a specific dependency grammar theory, such as the Prague School's Functional Generative Description, Meaning-Text Theory, or Hudson's Word Grammar. In practice most parsers constrain dependency structures to be tree-like structures such that each word has a single syntactic head, limiting diversity between annotation a bit; but while many dependency treebanks taking this format agree on how to analyze many syntactic constructions, there are still many constructions these treebanks analyze differently. See Figure 1 for a standard overview of clear and more difficult cases.

The difficult cases in Figure 1 are difficult for the following reason. In the easy cases morphosyntactic and semantic evidence cohere. Verbs govern subjects morpho-syntactically and seem semantically more important. In the difficult cases, however, morpho-syntactic evidence is *in conflict* with the semantic evidence. While auxiliary verbs have the same distribution as finite verbs in head position and share morpho-syntactic properties with them, and govern the infinite main verbs, main verbs seem semantically superior, expressing the main predicate. There may be distributional evidence that complementizers head verbs syntactically, but the verbs seem more important from a semantic point of view. Some authors have distinguished between the notion of functional (distributional) and substantive dependency heads (Ivanova et al., 2012).

Tree-to-dependency conversion schemes used to convert constituent-based treebanks into dependency-based ones also take different stands on the difficult cases. In this paper we consider four different conversion schemes: the Yamada-Matsumoto conversion scheme (Yamada) (Yamada and Matsumoto, 2003), the CoNLL (2007) format, the Stanford conversion scheme used in the English Web Treebank (Petrov and McDonald, 2012), and the LTH conversion scheme (Johansson and Nugues, 2007). The Yamada scheme can be replicated by running penn2malt.jar available at

`http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html`

We used Malt dependency labels (see online documentation). The Yamada scheme is an elaboration of the Collins scheme (Collins, 1999), which is not included in our experiments. The Stanford conversion scheme can be replicated using the Stanford converter available at

$FORM_1$	$FORM_2$	Yamada	CoNLL	Stanford	LTH
Auxiliary	Main verb	1	1	2	2
Complementizer	Verb	1	2	2	2
Coordinator	Conjuncts	2	1	2	2
Preposition	Nominal	1	1	1	2

Figure 2: Head decisions in conversions. Note: Yamada also differ from CoNLL in proper names.

	LTH	Stanford	Yamada
CoNLL	91.1%	89.7%	*92.7%*
LTH	-	89.7%	89.6%
Stanford	-	-	90.1%

Table 1: Unlabeled TED accuracies between conversion schemes

	LTH	Stanford	Yamada
CoNLL	6.05	5.70	*5.14*
LTH	-	6.85	7.06
Stanford	-	-	6.24

Table 2: L_1-distances between conversion schemes

`http://nlp.stanford.edu/software/stanford-dependencies.shtml`

The CoNLL 2007 conversion scheme can be obtained by running pennconverter.jar available at

`http://nlp.cs.lth.se/software/treebank_converter/`

with the 'conll07' flag set. The LTH conversion scheme can be obtained by running pennconverter.jar with the 'oldLTH' flag set.

We list the differences in Figure 2. It is clear from this list that the LTH scheme uses substantive heads more often than the other schemes. This makes sense as it was designed for downstream semantic role labeling (Johansson and Nugues, 2007). Somewhat surprisingly the scheme does not always lead to better semantic role labeling performance when relying on predicted syntactic parses, however (see Results).

3 Distance between schemes

We use three parse evaluation metrics to estimate distances between tree-to-dependency schemes.

The **NED scores** between pairs of gold annotated data using different conversion schemes give an empirical estimate of the coarser differences between the schemes. The NED scores between the Yamada scheme and the CoNLL and LTH schemes is 91.9-92.0%, for example, while the NED score between CoNLL and LTH is 93.9%. Interestingly, the NED between pairs of MATE outputs on our SMT tuning section (see Section 5.3) using different conversion schemes is 100% in all cases. This seems to indicate that differences in down-stream performance (see Table 4) should not be found in major theoretical differences, but rather small differences such as edge flippings and label granularity.

The **UA scores** (unlabeled attachment) punish edge flippings, and the fact that we observe low UA scores between conversion schemes support the above picture and also indicate that the differences are quite substantial. The overlap as measured by UA score between the Yamada and the CoNLL scheme is 78.9%, for example. These two schemes agree on the syntactic heads of about 4/5 words. The UA score between Yamada and LTH is 63.1%. Again we see that differences between gold annotated datasets using different conversion schemes are bigger than when comparing output pairs. **It seems that parser bias is canceling out differences between conversion schemes.**

We also report unlabeled **TED scores** (Tsarfaty et al., 2012) between output trees. TED is a three-step distance computation that first abstracts away from the directionality of head-dependent relations. In the second step, we separate the common and consistent parts of the two output trees, and in the third step we compute the tree-edit operations necessary to translate between the inconsistent subtrees. We use the SMT tune section again. The unlabeled TED accuracy between CoNLL and Yamada is 92.7%, and the L_1-distance is only 5.14. See tabels 1 and 2 for more results. The L_1-distances, together with the other metrics, suggest that CoNLL is more similar to the other conversion schemes than any other pair of schemes. See Figure 3.

	bl	Yamada	CoNLL	Stanford	LTH
DEPRELS	-	12	21	47	41
PTB-23 (LAS)	-	**88.99**	88.52	81.36*	87.52
PTB-23 (UAS)	-	90.21	90.12	84.22*	**90.29**
Neg: scope F_1	-	**81.27**	80.43	78.70	79.57
Neg: event F_1	-	76.19	72.90	73.15	**76.24**
Neg: full negation F_1	-	**67.94**	63.24	61.60	64.31
SentComp F_1	68.47	**72.07**	64.29	71.56	71.56
SMT-dev-Meteor	35.80	36.06	36.06	**36.16**	36.08
SMT-test-Meteor	37.25	37.48	37.50	**37.58**	37.51
SMT-dev-BLEU	13.66	**14.14**	14.09	14.04	14.06
SMT-test-BLEU	14.67	15.04	15.04	14.96	**15.11**
SRL-22-gold	-	81.35	83.22	**84.72**	84.01
SRL-23-gold	-	79.09	80.85	80.39	**82.01**
SRL-22-pred	-	74.41	76.22	**78.29**	66.32
SRL-23-pred	-	73.42	74.34	**75.80**	64.06
bitterlemons.org	96.08	**97.06**	95.58	96.08	96.57

Table 4: Results. *: Low parsing results on PTB-23 using Stanford are explained by changes between the PTB-III and the Ontonotes 4.0 release of the English Treebank.

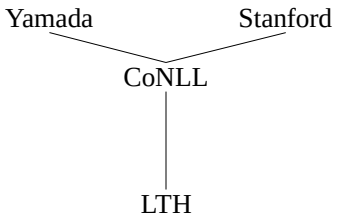

Figure 3: L_1-distances between conversion schemes

	gold		smt-tune	
	w-ppl	d-ppl	w-ppl	d-ppl
CoNLL	243.3	545.6	273.5	430.4
LTH	243.3	536.2	273.5	428.0
Stanford	243.3	541.9	273.5	425.4
Yamada	243.3	**520.8**	273.5	426.5

Table 3: Derivational perplexity of converted treebanks

4 Learnability and derivational perplexity

Schwartz et al. (2012) study the learnability of different conversion schemes, by relating choises concerning head status to parser performance. How does making prepositions head their complements affect parser performance, for example? The first two lines in Table 4 suggest, in line with Schwartz et al. (2012), that Yamada is more learnable than the other three schemes.

The derivational perplexity of a treebank T (Søgaard and Haulrich, 2010) is defined as the perplexity (per word) of the derivation language of $f(T)$, where $f(T)$ is the canonical derivation orders of the dependency trees in T: The derivation language of $f(T)$ is the set of strings $\sigma : w_1...w_n$ such that for any w_i, w_j $w_i \prec w_j$ in dependency structure d if w_i was attached to d in $f(T)$ prior to the attachment of w_j, according to their canonical derivation. Canonical derivations are also used to train transition-based dependency parsers, and we use the arc-eager algorithm in our experiments below (Nivre et al., 2007). We use a trigram language model with Knesser-Ney smoothing, replicating the setup in Søgaard and Haulrich (2010).

The results in Table 3 seem to indicate that the Yamada scheme is more learnable than the more recent ones, but again the differences seemed to be cancelled out by parser bias, and differences in derivational perplexity of the output (again using the tuning section of the SMT data) become insignificant. The reason for the absolute drop in numbers, as well as the drop in differences, is

probably the shorter sentence length and more straight-forward syntax often associated with spoken language such as the parliament discussions in the Europarl data.

Learnability vs. derivational perplexity. Søgaard and Haulrich (2010) show that derivational perplexity correlates well with performance in multi-lingual parsing (using data from the CoNLL-X and CoNLL 2007 shared tasks), and therefore also with learnability (Schwartz et al., 2012). We think derivational perplexity is a better measure of performance robustness for two reasons: because it is a parser-independent metric, (i) derivational perplexities are not influenced by regularization, and (ii) derivational perplexities are not influenced by different parser biases. Changing a parameter in a tree-to-dependency conversion scheme may affect parser performance for several reasons: It is well known that adding features that never fire sometimes leads to improved performance with state-of-the-art parsers such as the MaltParser (Nivre et al., 2007), because regularization is sensitive to the total number of features, and it is not always clear whether a choice of head status leads to improved performance because the head choice is more learnable, or because it has an effect on regularization. Finally, annotation interacts with parser bias. Some choices of head status may be easy to learn for transition-based dependency parsers, but comparatively harder for graph-based ones. See McDonald and Nivre (2007) for an analysis of the biases of transition-based and graph-based dependency parsers. Since derivational perplexity is parser-independent we are not sensitive to regularization or parser bias, only to the choice of canonical derivation scheme.

5 Downstream performance

Dependency parsing has proven useful for a wide range of NLP applications, including statistical machine translation (Galley and Manning, 2009; Xu et al., 2009; Elming and Haulrich, 2011) and sentiment analysis (Joshi and Penstein-Rose, 2009; Johansson and Moschitti, 2010). Below we introduce five NLP applications where dependency parsing has been succesfully applied: negation resolution, semantic role labeling, statistical machine translation, sentence compression and perspective classification. We will then report on evaluations of the downstream effects of the four conversion schemes in these five applications, first published in Elming et al. (2013).

In the five applications we use syntactic features in slightly different ways. While our statistical machine translation and sentence compression systems use dependency relations as additional information about words and *on a par* with POS, our negation resolution system uses dependency paths, conditioning decisions on both dependency arcs and labels. In perspective classification, we use dependency triples (e.g. SUBJ(John, snore)) as features, while the semantic role labeling system conditions on a lot of information, including the word form of the head, the dependent and the argument candidates, the concatenation of the dependency labels of the predicate, and the labeled dependency relations between predicate and its head, its arguments, dependents or siblings.

5.1 Negation resolution

Negation resolution (NR) is the task of finding negation cues, e.g. the word *not*, and determining their *scope*, i.e. the tokens they affect. NR has recently seen considerable interest in the NLP community (Morante and Sporleder, 2012; Velldal et al., 2012) and was the topic of the 2012 *SEM shared task (Morante and Blanco, 2012).

The data set used in this work, the Conan Doyle corpus (CD),[1] was released in conjunction with the *SEM shared task. The annotations in CD extend on cues and scopes by introducing annotations for in-scope events that are negated in factual contexts. The following is an example from the corpus showing the annotations for cues (bold), scopes (underlined) and negated events (italicized):

(1) Since we have been so
un*fortunate* as to miss him [...]

CD-style scopes can be discontinuous and overlapping. Events are a portion of the scope that is semantically negated, with its truth value reversed by the negation cue.

The NR system used in this work (Lapponi et al., 2012), one of the best performing systems in the *SEM shared task, is a CRF model for scope resolution that relies heavily on features extracted from dependency graphs. The feature model contains token distance, direction, n-grams of word

[1] http://www.clips.ua.ac.be/sem2012-st-neg/data.html

REFERENCE:	Zum Glück kam ich beim Strassenbahnfahren an die richtige Stelle .
SOURCE:	Luckily , on the way to the tram , I found the right place .
Yamada:	Glücklicherweise hat auf dem Weg zur S-Bahn , stellte ich fest , dass der richtige Ort .
CoNLL:	Glücklicherweise hat auf dem Weg zur S-Bahn , stellte ich fest , dass der richtige Ort .
Stanford:	Zum Glück fand ich auf dem Weg zur S-Bahn , am richtigen Platz .
LTH:	Zum Glück fand ich auf dem Weg zur S-Bahn , am richtigen Platz .
BASELINE:	Zum Glück hat auf dem Weg zur S-Bahn , ich fand den richtigen Platz .

Figure 4: Examples of SMT output.

ORIGINAL:	* 68000 sweden ab of uppsala , sweden , introduced the teleserve , an integrated answering machine and voice-message handler that links a macintosh to touch-tone phones .
BASELINE:	68000 sweden ab introduced the teleserve an integrated answering machine and voice-message handler .
Yamada	68000 sweden ab introduced the teleserve integrated answering machine and voice-message handler .
CoNLL	68000 sweden ab **sweden** introduced the teleserve integrated answering machine and voice-message handler .
Stanford	68000 sweden ab introduced the teleserve integrated answering machine and voice-message handler .
LTH	68000 sweden ab introduced the teleserve **an** integrated answering machine and voice-message handler .
HUMAN:	68000 sweden ab introduced the teleserve integrated answering machine and voice-message handler .

Figure 5: Examples of sentence compression output.

Syntactic	constituent
	dependency relation
	parent head POS
	grand parent head POS
	word form+dependency relation
	POS+dependency relation
Cue-dependent	directed dependency distance
	bidirectional dependency distance
	dependency path
	lexicalized dependency path

Figure 6: Features used to train the conditional random field models

forms, lemmas, POS and combinations thereof, as well as the syntactic features presented in Figure 6. The results in our experiments are obtained from configurations that differ only in terms of tree-to-dependency conversions, and are trained on the training set and tested on the development set of CD. Since the negation cue classification component of the system does not rely on dependency features at all, the models are tested using gold cues.

Table 4 shows F_1 scores for scopes, events and full negations, where a true positive correctly assigns both scope tokens and events to the rightful cue. The scores are produced using the evaluation script provided by the *SEM organizers.

5.2 Semantic role labeling

Semantic role labeling (SRL) is the attempt to determine semantic predicates in running text and label their arguments with semantic roles. In our experiments we have reproduced the second best-performing system in the CoNLL 2008 shared task in syntactic and semantic parsing (Johansson and Nugues, 2008).[2]

The English training data for the CoNLL 2008 shared task were obtained from PropBank and NomBank. For licensing reasons, we used OntoNotes 4.0, which includes PropBank, but not NomBank. This means that our system is only trained to classify verbal predicates. We used the Clearparser conversion tool[3] to convert the OntoNotes 4.0 and subsequently supplied syntactic dependency trees using our different conversion schemes. We rely on gold standard argument identification and focus solely on the performance metric semantic labeled F1.

5.3 Statistical machine translation

The effect of the different conversion schemes was also evaluated on SMT. We used the *reordering by parsing* framework described by Elming and Haulrich (2011). This approach integrates a syntactically informed reordering model into a phrase-based SMT system. The model learns to predict the word order of the translation based on source sentence information such as syntactic

[2] http://nlp.cs.lth.se/software/semantic_parsing:_propbank_nombank_frames

[3] http://code.google.com/p/clearparser/

dependency relations. Syntax-informed SMT is known to be useful for translating between languages with different word orders (Galley and Manning, 2009; Xu et al., 2009), e.g. English and German.

The baseline SMT system is created as described in the guidelines from the original shared task.[4] Only modifications are that we use truecasing instead of lowercasing and recasing, and allow training sentences of up to 80 words. We used data from the English-German restricted task: $\sim$3M parallel words of news, $\sim$46M parallel words of Europarl, and $\sim$309M words of monolingual Europarl and news. We use newstest2008 for tuning, newstest2009 for development, and newstest2010 for testing. Distortion limit was set to 10, which is also where the baseline system performed best. The phrase table and the lexical reordering model is trained on the union of all parallel data with a max phrase length of 7, and the 5-gram language model is trained on the entire monolingual data set.

We test four different experimental systems that only differ with the baseline in the addition of a syntactically informed reordering model. The baseline system was one of the tied best performing system in the WMT 2011 shared task on this dataset. The four experimental systems have reordering models that are trained on the first 25,000 sentences of the parallel news data that have been parsed with each of the tree-to-dependency conversion schemes. The reordering models condition reordering on the word forms, POS, and syntactic dependency relations of the words to be reordered, as described in Elming and Haulrich (2011). The paper shows that while reordering by parsing leads to significant improvements in standard metrics such as BLEU (Papineni et al., 2002) and METEOR (Lavie and Agarwal, 2007), improvements are more spelled out with human judgements. All SMT results reported below are averages based on 5 MERT runs following Clark et al. (2011).

5.4 Sentence compression

Sentence compression is a restricted form of sentence simplification with numerous usages, including text simplification, summarization and recognizing textual entailment. The most commonly used dataset in the literature is the Ziff-Davis corpus.[5] A widely used baseline for sentence compression experiments is the two models introduced in Knight and Marcu (2002): the noisy-channel model and the decision tree-based model. Both are tree-based methods that find the most likely compressed syntactic tree and outputs the yield of this tree. McDonald et al. (2006) instead use syntactic features to directly find the most likely compressed sentence.

Here we learn a discriminative HMM model (Collins, 2002) of sentence compression using MIRA (Crammer and Singer, 2003), comparable to previously explored models of noun phrase chunking. Our model is thus neither tree-based nor sentence-based. Instead we think of sentence compression as a sequence labeling problem. We compare a model informed by word forms and predicted POS with models also informed by predicted dependency labels. The baseline feature model conditions emission probabilities on word forms and POS using a ± 2 window and combinations thereoff. The augmented syntactic feature model simply adds dependency labels within the same window.

5.5 Perspective classification

Finally, we include a document classification dataset from Lin and Hauptmann (2006).[6] The dataset consists of blog posts posted at bitterlemons.org by Israelis and Palestinians. The bitterlemons.org website is set up to "contribute to mutual understanding through the open exchange of ideas." In the dataset, each blog post is labeled as either Israeli or Palestinian. Our baseline model is just a standard bag-of-words model, and the system adds dependency triplets to the bag-of-words model in a way similar to Joshi and Penstein-Rose (2009). We do not remove stop words, since perspective classification is similar to authorship attribution, where stop words are known to be informative. We evaluate performance doing cross-validation over the official training data, setting the parameters of our learning algorithm for each fold doing cross-validation over the actual training data. We used soft-margin support vector machine learning (Cortes and Vapnik, 1995), tuning the kernel (linear or polynomial with degree 3) and $C = \{0.1, 1, 5, 10\}$.

[4] http://www.statmt.org/wmt11/translation-task.html

[5] LDC Catalog No.: LDC93T3A.

[6] https://sites.google.com/site/weihaolinatcmu/data

5.6 Results

Our results are presented in Table 4. The parsing results are obtained relying on predicted POS rather than, as often done in the dependency parsing literature, relying on gold-standard POS. Note that they comply with the result in Schwartz et al. (2012) that Yamada annotation is more easily learnable.

The **negation resolution** results are significantly better using syntactic features in Yamada annotation. It is not surprising that a syntactically oriented conversion scheme performs well in this task.

The case-sensitive BLEU evaluation of the **SMT** systems indicates that choice of conversion scheme has no significant impact on overall performance. The difference to the baseline system is significant ($p < 0.01$), showing that the reordering model leads to improvement using any of the schemes. Differences between schemes are insignificant. The reason probably is that long-distance differences between the schemes are cancelled out by parser bias. However, the conversion schemes lead to very different translations. This can be seen, for example, by the fact that the (normalized) string edit distance between translations of different syntactically informed SMT systems is 12% higher than within each system (across different MERT optimizations).

The reordering approach puts a lot of weight on the syntactic dependency relations. As a consequence, the number of relation types used in the conversion schemes proves important. Consider the example in Figure 4. German requires the verb in second position (V2), which is picked up by the Stanford and LTH systems. Interestingly, the four schemes produce virtually identical structures for the source sentence, but they differ in their labeling. Where CoNLL and Yamada use the same relation for the first two constituents (ADV and vMOD, respectively), Stanford and LTH distinguish between them (ADVMOD/PREP and ADV/LOC). This distinction may be what enables learning V2 translations, since the model may learn to move the verb after the sentence adverbial. In the other schemes, sentence adverbials are not distinguished from locational adverbials. Generally, Stanford and LTH have more than twice as many relation types as the other schemes.

The schemes Stanford and LTH lead to better **SRL** performance than CoNLL and Yamada when relying on gold-standard syntactic dependency trees. This supports the claims put forward in Johansson and Nugues (2007). These annotations also happen to use a larger set of dependency labels, however, and syntactic structures may be harder to reconstruct, as reflected by labeled attachment scores (LAS) in syntactic parsing. The biggest drop in SRL performance going from gold-standard to predicted syntactic trees is clearly for the LTH scheme, at an average 17.8% absolute loss (Yamada 5.8%; CoNLL 6.8%; Stanford 5.5%; LTH 17.8%).

The Stanford scheme resembles LTH in most respects, but in preposition-noun dependencies it marks the preposition as the head rather than the noun. This is an important difference for SRL, because semantic arguments are often nouns embedded in prepositional phrases, like agents in passive constructions. It may also be that the difference in performance is simply explained by the syntactic analysis of prepositional phrases being easier to reconstruct.

The **sentence compression** results are generally much better than the models proposed in Knight and Marcu (2002). Their noisy channel model obtains an F_1 compression score of 14.58%, whereas the decision tree-based model obtains an F_1 compression score of 31.71%. While F_1 scores should be complemented by human judgements, as there are typically many good sentence compressions of any source sentence, we believe that error reductions of more than 50% indicate that the models used here (though previously unexplored in the literature) are fully competitive with state-of-the-art models.

We also see that the models using syntactic features perform better than our baseline model, except for the model using CoNLL dependency annotation. This may be surprising to some, since distributional information is often considered important in sentence compression (Knight and Marcu, 2002). Some output examples are presented in Figure 5. Unsurprisingly, it is seen that the baseline model produces grammatically incorrect output, and that most of our syntactic models correct the error leading to ungrammaticality. The model using Stanford annotation is an exception. We also see that CoNLL introduces another error. We believe that this is due to the way the CoNLL tree-to-dependency conversion scheme handles coordination. While the word

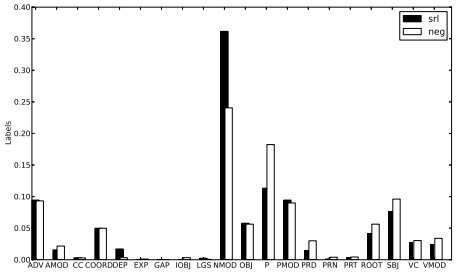

Figure 7: Distributions of dependency labels in the Yamada-Matsumoto scheme

Sweden is not coordinated, it occurs in a context, surrounded by commas, that is very similar to coordinated items.

In **perspective classification** we see that syntactic features based on Yamada and LTH annotations lead to improvements, with Yamada leading to slightly better results than LTH. The fact that a syntactically oriented conversion scheme leads to the best results may reflect that perspective classification, like authorship attribution, is less about content than stylistics.

While LTH seems to lead to the overall best results, we stress the fact that the five tasks considered here are incommensurable. What is more interesting is that, task to task, results are so different. The semantically oriented conversion schemes, Stanford and LTH, lead to the best results in SRL, but with a significant drop for LTH when relying on predicted parses, while the Yamada scheme is competitive in the other four tasks. This may be because distributional information is more important in these tasks than in SRL.

The distribution of dependency labels seems relatively stable across applications, but differences in data may of course also affect the usefulness of different annotations. Note that CoNLL leads to very good results for negation resolution, but bad results for SRL. See Figure 7 for the distribution of labels in the CoNLL conversion scheme on the SRL and negation scope resolution data. Many differences relate to differences in sentence length. The negation resolution data is literary text with shorter sentences, which therefore uses more punctuation and has more root dependencies than newspaper articles. On the other hand we do see very few predicate dependencies in the SRL data. This may affect down-stream results when classifying verbal predicates in SRL. We also note that the number of dependency labels have less impact on results in general than we would have expected. The number of dependency labels and the lack of support for some of them may explain the drop with predicted syntactic parses in our SRL results, but generally we obtain our best results with Yamada and LTH annotations, which have 12 and 41 dependency labels, respectively.

6 Discussion and conclusion

In our experiments we made several observations. Available tree-to-dependency conversion schemes are very different. On the other hand we saw that many of the non-local differences between conversion schemes are not learned by state-of-the-art parsers, making parser output across conversion schemes less different from gold annotations. This suggests that only more local differences are important for downstream performance, which may also explain the small differences observed in our SMT experiments.

While the CoNLL scheme is very learnable (Schwartz et al., 2012), second to Yamada, downstream performance suggest that it is a suboptimal conversion scheme. The Yamada scheme is both very learnable (1st), leads to very good downstream performance (1st or 2nd in 4/5 downstream applications), and it has low derivational perplexity. We have argued that this is a better metric for performance robustness than learnability.

Note that our results extend beyond dependency parsing. Ivanova et al. (2012) show that converted dependency structures bear similarities to both Stanford dependencies and DELPH-IN syntactic derivation structures, but they are not explicit about what conversion scheme was used.

In future work we would like to combine the different methodologies discussed here to be able to learn robust annotation for given end applications that optimize end performance.

References

Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In *COLING*.

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: controlling for optimizer instability. In *ACL*.

Mike Collins. 1999. *Head-driven statistical models*

for natural language parsing. Ph.D. thesis, University of Pennsylvania.

Michael Collins. 2002. Discriminative training methods for Hidden Markov Models. In *EMNLP*.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20(3):273–297.

Koby Crammer and Yoram Singer. 2003. Ultraconservative algorithms for multiclass problems. In *JMLR*.

Jakob Elming and Martin Haulrich. 2011. Reordering by parsing. In *Proceedings of International Workshop on Using Linguistic Information for Hybrid Machine Translation (LIHMT-2011)*.

Jakob Elming, Anders Johannsen, Sigrid Klerke, Emanuele Lapponi, Hector Martinez Alonso, and Anders Søgaard. 2013. Down-stream effects of tree-to-dependency conversions. In *NAACL*.

Michel Galley and Christopher Manning. 2009. Quadratic-time dependency parsing for machine translation. In *ACL*.

Angelina Ivanova, Stephan Oepen, Lilja Øvrelid, and Dan Flickinger. 2012. Who did what to whom? a contrastive study of syntactico-semantic dependencies. In *LAW*.

Richard Johansson and Alessandro Moschitti. 2010. Syntactic and semantic structure for opinion expression detection. In *CoNLL*.

Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In *NODALIDA*.

Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic-semantic analysis with propbank and nombank. In *CoNLL*.

Richard Johansson. 2013. Training parsers on incompatible treebanks. In *NAACL*.

Mahesh Joshi and Carolyn Penstein-Rose. 2009. Generalizing dependency features for opinion mining. In *ACL*.

Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: a probabilistic approach to sentence compression. *Artificial Intelligence*, 139:91–107.

Emanuele Lapponi, Erik Velldal, Lilja Øvrelid, and Jonathon Read. 2012. UiO2: Sequence-labeling negation using dependency features. In **SEM*.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: an automatic metric for mt evaluation with high levels of correlation with human judgments. In *WMT*.

Wei-Hao Lin and Alexander Hauptmann. 2006. Are these documents written from different perspectives? In *COLING-ACL*.

Mitchell Marcus, Mary Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Ryan McDonald and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsers. In *EMNLP-CoNLL*.

Ryan McDonald. 2006. Discriminative sentence compression with soft syntactic evidence. In *EACL*.

Roser Morante and Eduardo Blanco. 2012. *sem 2012 shared task: Resolving the scope and focus of negation. In **SEM*.

Roser Morante and Caroline Sporleder. 2012. Modality and negation: An introduction to the special issue. *Computational linguistics*, 38(2):223–260.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 Shared Task on Dependency Parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic.

Kishore Papineni, Salim Roukus, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania.

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 Shared Task on Parsing the Web. In *Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL)*.

Roy Schwartz, Omri Abend, and Ari Rappoport. 2012. Learnability-based syntactic annotation design. In *COLING*.

Anders Søgaard and Martin Haulrich. 2010. On the derivation perplexity of treebanks. In *TLT*.

Reut Tsarfaty, Joakim Nivre, and Evelina Andersson. 2012. Cross-framework evaluation for statistical parsing. In *EACL*.

Erik Velldal, Lilja Øvrelid, Jonathon Read, and Stephan Oepen. 2012. Speculation and negation: Rules, rankers, and the role of synta. *Computational linguistics*, 38(2):369–410.

Peng Xu, Jaeho Kang, Michael Ringgaard, and Franz Och. 2009. Using a dependency parser to improve SMT for subject-object-verb languages. In *Proceedings of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT) 2009*, pages 245–253, Boulder, Colorado.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the 8th International Workshop on Parsing Technologies*, pages 195–206, Nancy, France.