Coordinate Constructions in English Enhanced Universal Dependencies: Analysis and Computational Modeling

Stefan Grünewald[1,2] **Prisca Piccirilli**[1,2] **Annemarie Friedrich**[2]
[1]Institut für Maschinelle Sprachverarbeitung, University of Stuttgart
[2]Bosch Center for Artificial Intelligence, Renningen, Germany
stefan.gruenewald|annemarie.friedrich@de.bosch.com
piccirpa@ims.uni-stuttgart.de

Abstract

In this paper, we address the representation of coordinate constructions in Enhanced Universal Dependencies (UD), where relevant dependency links are propagated from conjunction heads to other conjuncts. English treebanks for enhanced UD have been created from gold basic dependencies using a heuristic rule-based converter, which propagates only core arguments. With the aim of determining which set of links *should* be propagated from a semantic perspective, we create a large-scale dataset of manually edited syntax graphs. We identify several systematic errors in the original data, and propose to also propagate adjuncts. We observe high inter-annotator agreement for this semantic annotation task. Using our new manually verified dataset, we perform the first principled comparison of rule-based and (partially novel) machine-learning based methods for conjunction propagation for English. We show that learning propagation rules is more effective than hand-designing heuristic rules. When using automatic parses, our neural graph-parser based edge predictor outperforms the currently predominant pipelines using a basic-layer tree parser plus converters.

1 Introduction

The Universal Dependencies (UD) formalism (de Marneffe et al., 2014) is a framework for representing syntactic dependencies between words, prioritizing links between content words. UD parses provide two levels of analysis. *Basic* dependencies form standard syntactic dependency trees in which each node has exactly one governor (black links on top in Figure 1). *Enhanced* dependencies (Schuster and Manning, 2016) are extensions of these trees including additional relations (blue links below sentence) with the aim of representing linguistic phenomena such as coordination, control, or relative clauses. They have been shown to provide valuable

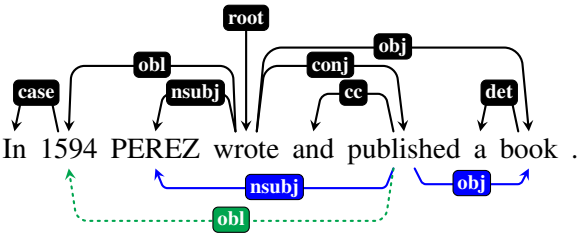


Figure 1: UD **basic** (top) and **enhanced** (bottom) dependencies. **Green dotted link:** proposed addition.

input for information extraction tasks (Schuster et al., 2017). One of the most frequent phenomena addressed by enhanced UD is coordination. In the English Web Treebank (EWT), more than 15% of all sentences contain conjoined verbs. Hence, a good representation of coordination clearly is crucial for downstream tasks. For example, in Figure 1, the enhanced layer explicitly captures that the arguments of the predicate "wrote" also fill the corresponding slots of "published," which is highly relevant for natural language understanding tasks.

In many cases, enhanced representations can be derived from the gold basic layer in a rule-based fashion (Schuster and Manning, 2016). The currently available English enhanced UD treebanks have been created by applying such a converter. However, we are not aware of a large study regarding their correctness and completeness. Focusing on precision, the converter only propagates core arguments. In this paper, we take a complementary approach, performing a large-scale annotation study in order to determine which set of links *should* be propagated from a semantic perspective. On a new dataset of 1,417 sentences from the EWT containing conjoined verbs, we verify and if necessary modify/extend the links involved in coordinate constructions. We argue that adjuncts such as obliques should in fact be propagated at times, e.g., in Figure 1, the additional (green dotted) link that

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 795–809
April 19 - 23, 2021.

we propose to add facilitates answering questions like "When was the book published?". To the best of our knowledge, our work constitutes the first large-scale annotation effort of this kind.

On the basis of our new dataset, we make the following contributions. First, we estimate the degree of **correctness and completeness of the rule-based converter/existing treebanks**. We find that the converter usually generates correct graphs when applied on gold basic trees, with some notable exceptions involving non-parallel syntactic constructions (e.g., conjuncts having different voice or mood). In addition, the converter does not propagate links correctly in presence of multiple interacting conjunctions. Our inter-anntotator agreement study shows high overlap for propagation decisions, with F1 between pairs of annotators of about 0.9 on average and around 0.75 for obliques.

Second, we address the question of **how to create high-quality treebanks** for enhanced UD from gold basic dependencies, again focusing on coordinate constructions. Based on the findings of our corpus study, we improve the rule-based converter by Schuster and Manning (2016). We also compare machine-learning (ML) based conjunction propagation classifiers in the form of (a) SVM-based classifiers as previously used for Finnish, Swedish and Italian (Nyblom et al., 2013; Nivre et al., 2018), and (b) a novel neural approach integrating tree- and RoBERTa-based features. We find that all systems mostly rely on tree-based features, but contextual embeddings also provide useful information. Performance on propagation decisions has promising F1 around 0.9, already similar to human agreement. ML-based classifiers outperform the rule-based converters on the EWT test set.

Third, we compare methods for **extracting propagated dependencies in an automatic parsing setting.** The currently predominant approach is to run a basic-layer tree parser and then the same converter that has been used for gold standard construction. We propose to use a neural graph-parser based edge predictor with an architecture similar to Dozat and Manning (2018) instead, and show that this approach outperforms pipelines by around 9 points F1 on propagating links in conjunctions.

In sum, our contributions include: (1) a manually curated large-scale dataset of 1,417 sentences addressing semantically motivated correct and complete conjunction propagation in enhanced UD; (2) the proposal of novel neural approaches to conjunction propagation; and (3) experimental evidence that these models outperform rule- and pipeline-based approaches in both gold standard treebank enhancing and automatic parsing settings. To the best of our knowledge, our work constitutes the first principled comparison of various approaches to propagating conjunctions in enhanced UD on manually corrected gold standard data for English. Both our model implementations and the dataset are freely available.[1] We will contribute our changes to the EWT corpus to the next UD release.

2 Related Work

Coordinate Constructions in UD are represented using the *conj* relation, with the first conjunct being the head to which all dependencies of the phrase are attached (see Figure 1). In the basic layer, all governors and dependents of a conjoined phrase are attached to the first conjunct. In the enhanced layer, relations are propagated to the dependent if suggested by the semantics of the sentence.[2] Schuster and Manning (2016) present an algorithm for creating enhanced dependencies automatically based on the basic layer. While it propagates links with high precision, it propagates *only* core arguments by design (see Appendix A). In addition, it is highly reliant on correct basic dependencies (see Sec. 5).

Conjunction propagation classifiers. Nyblom et al. (2013) present an SVM-based approach for enhancing Finnish syntax trees. They observe high performance on conjunction propagation when operating on gold basic trees, but markedly worse results when using automatic parser output. Nivre et al. (2018) evaluate a similar approach for Swedish and Italian. We show that their approach also works well for English, and extend it with neural models and contextualized word embeddings. Simi and Montemagni (2018) de-lexicalize their rule-based converter developed for Italian, showing that their language-independent system also correctly produces most of the propagations for English. However, they evaluate on EWT, which is itself the result of a rule-based system. In contrast, we evaluate on manually checked gold data.

For the related task of dealing with **gapping** constructions such as "Paul likes coffee and Mary tea,"

[1] https://github.com/boschresearch/coordinate_constructions_english_enhanced_ud_eacl2021

[2] https://universaldependencies.org/u/overview/enhanced-syntax.html

Schuster et al. (2018) reconstruct elided predicates by first parsing into an intermediate representation and then applying either a rule-based or an ML-based algorithm to copy over lexical material. We here focus on dependency propagation and operate on gold tokens as annotated in the enhanced UD treebanks, which already include traces. Other related work exists in the area of manual and rule-based error correction on UD treebanks (Wisniewski, 2018; Alzetta et al., 2018).

There is still little published work regarding **fully automatic enhanced UD parsing**, however, the topic has recently been addressed by the IWPT 2020 Shared Task (Bouma et al., 2020). Among the top-performing systems, several approaches first parse into basic UD and then added transformation rules (e.g., Heinecke, 2020; Dehouck et al., 2020). Others directly employ graph parsing techniques (e.g., Wang et al., 2020; He and Choi, 2020; Hershcovich et al., 2020). The overall winner TurkuNLP (Kanerva et al., 2020) transforms enhanced UD into a tree format and then makes use of UDify (Kondratyuk and Straka, 2019). In addition, much work exists on **semantic dependency parsing** (SDP, Oepen et al., 2014, 2015; May and Priyadarshi, 2017). These works differ from UD-based approaches as the respective formalisms represent meaning less close to syntactic structure, thus not requiring propagation. From a modeling point of view, our work is most similar to that of Grünewald and Friedrich (2020), who also use a graph-based biaffine architecture for enhanced UD parsing, and to that of Dozat and Manning (2018), who achieve state-of-the-art results for SDP.

3 Coordinate Constructions Dataset

In this section, we describe our creation of our manually created dataset and analyse the results.

	conj. sentences	edited
train	1,926	999
dev	222	222
test	196	196
total	2,344	1,417

Table 1: Coordinate constructions dataset statistics. **conj. sentences**: sentences in EWT containing verb phrase conjunctions; we edited 60% of these.

3.1 Data

Our dataset consists of 1,417 sentences collected from EWT,[3] containing data from five genres of web media (weblogs, newsgroups, emails, reviews, and Yahoo! answers).[4] The basic dependencies of this UD gold standard have been derived from the original Stanford dependencies (de Marneffe et al., 2006) and were then hand-corrected. The enhanced layer has been created using the automatic converter (Schuster and Manning, 2016, see Appendix A). We retrieve all sentences containing at least one *conj* link between two verbs. More than 15% of all sentences in EWT contain conjoined verbs. Out of these sentences, we edit all sentences of the dev and test sets, and 999 sentences of the training set, amounting to more than 60% of all relevant sentences in EWT (see Table 1). The careful curation of each sentence took around 10 minutes on average, amounting to a total annotation effort of around 240 hours (total costs ca. $4,750). We exclude 18 sentences when reporting our statistics: In 12 cases, the *conj* relation is annotated wrongly in the basic layer and six sentences contain syntactically non-standard English.[5]

3.2 Annotation Methodology

The manual corrections of the treebank were performed by a French native speaker with an extensive background in linguistics. The annotation project involved regular discussions among all authors to decide on uncertain cases and to ensure consistency. Additionally, in case of doubt, an English native speaker with an extensive linguistics background was consulted. Dependencies were checked carefully sentence-wise using the ConLL-U-Editor tool (Heinecke, 2019). If necessary, the full document was consulted to make sure interpretations were correct in context.

Annotation Guidelines. We verify and modify all links involved in coordinate constructions including conjoined verbs, but also noun or adjectival phrases. First, we make sure that the automatically constructed enhanced representations adhere to the official guidelines for enhanced UD (see Sec. 2), propagating heads and dependents of conjuncts if

[3] Linguistic Data Consortium LDC2012T13.

[4] https://universaldependencies.org/treebanks/en_ewt/index.html

[5] Such as "i want to be able to use it in my car, out n about etc...i guess like an iphone, but thats later on and ,i know what they are so no suggestions on just goin out to buy one im talking about right now just for an ipod??" (EWT dev set)

	A	B	C
A	-	90.1	94.9
B	95.2	-	97.2
C	80.5	77.9	-

Table 2: Inter-annotator agreement on propagated links for 100 sentences: **precision** when treating the row annotator as gold standard (or, equivalently, **recall** when treating the column annotator as gold standard).

the interpretation of the sentence suggests additional syntactic relations between words.

As each verb may also have its own complements, this task requires a semantic interpretation leveraging context and knowledge about selectional preferences. If an ambiguity has already been resolved in the basic layer,[6] we follow this interpretation unless obviously wrong. Second, we propose to also propagate non-core dependents such as *obl*, *advcl* and *advmod* if suggested by semantics, an annotation task similar to prepositional phrase attachment resolution. We only propagate such links if the adjunct clearly modifies each conjunct (as in Figure 1). Finally, we extend the attachment of relative pronouns (*ref*) to all antecedents if involved in coordinations. We focus on propagating dependencies between content words, not propagating relations such as *aux* or *cop*, which could be handled as traces.

Inter-annotator agreement study. We sampled 100 sentences, half of them from cases where the primary annotator had judged the original version to be correct, and half of them cases that included modifications. This sample was blindly re-annotated by two secondary annotators, both German native speakers with an extensive computational linguistics background. Table 2 shows agreement in terms of precision and recall on the set of dependencies resulting from conjunction propagation, i.e., the links involved in conjunctions that are present in the enhanced layer but not in the basic layer. For a formal definition, see Appendix B. Agreement is generally high, particularly between annotators A and B. Annotator C was more conservative in propagating links, especially in generally ambiguous cases. However, the links that C propagates are also propagated by A and B. Pairwise agreement was high on *nsubj*, *obj* and *xcomp*. Modifier clauses (*acl*, *advcl*) and adverbials (*advmod*) were common sources of disagreement, indicating the more ambiguous nature of these propagations. Pairwise scores and more details can be found in Appendix B.

[6] For example, in "She was reading or watching a movie," "movie" is attached to the second conjunct "watching" in the basic layer, hence resolving the syntactic ambiguity.

3.3 Analysis and Discussion

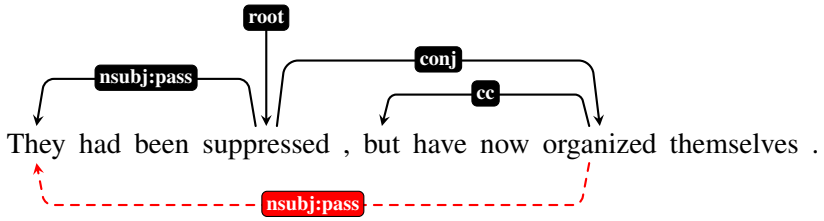
In this section, we analyse and discuss the modifications made to the original treebanks.

Quantitative Analysis of Changes. Table 3 presents the numbers of dependency relations that have been added and removed in coordinate constructions in the enhanced layer. More specifically, we consider only the set of links not present in the basic tree and count modifications regarding links starting or ending at conjuncts.[7] Counts for coarse-grained labels (e.g., *nmod*) include all subtypes (e.g., *nmod:for*) not explicitly listed in the table. During our manual correction of the treebank, around 15% of the total enhanced links involved in conjoined phrases were added and about 3% were removed. This confirms that the converter by Schuster and Manning (2016) is optimized for precision rather than recall, though our additions of course include labels that the converter does not address. Note that in these cases, removed relations in Table 3 are caused by fixes regarding attachment in the basic layer, whose errors had been propagated to the enhanced layer. In total, we fixed errors in 57 sentences in the basic layer. In 42 of these, this led to changes in the enhanced layer.

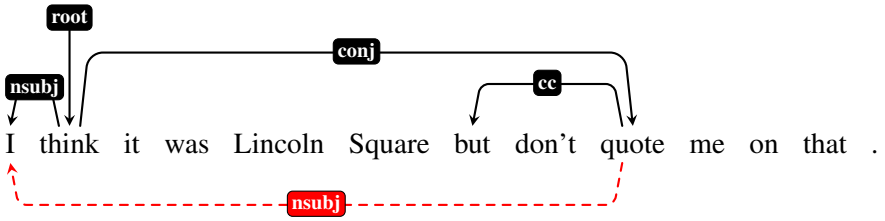
Linguistic Analysis of Changes. One systematic error involves **links to subjects in passive constructions**: 18 out of 225 *nsubj:pass* links were actually wrongly propagated. All of them have been changed to *nsubj*. The reason is that the converter automatically propagates an *nsubj:pass* link if the first conjoined verb is in the passive form, as, e.g., in "These Shiite movements had been suppressed by Saddam Hussein's regime, but have now organized and armed themselves" (see Figure 2a). Another common error (occurring 12 times) is the propagation of the first conjoined verb's subject to the second verb, even though the latter is in **imperative mood**, as, e.g., in "I think it was the Lincoln Square area but don't quote me on that" (see Figure 2b).

In sentences containing **multiple coordinate constructions**, such as "Dr. Fortier and his girlfriend lashed two canoes together and paddled

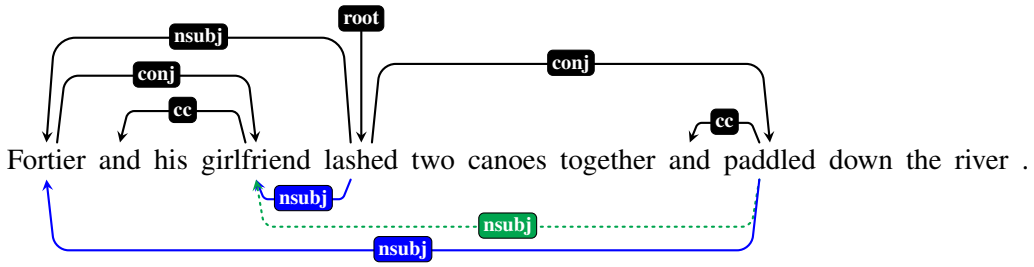
[7] We made some additional fixes (not necessarily related to coordinations) to the original treebank, see Appendix C.



(a) Passive voice: *nsubj:pass* links should not be propagated verbatim if the second conjunct is in active voice.



(b) Imperative mood: *nsubj* links should not be propagated if the second conjunct is in imperative mood.



(c) Multiple coordinate constructions: *nsubj* links should be propagated between the second conjuncts of each coordination.

Figure 2: Systematic errors found in the automatic propagation of dependencies. (Only coordination-relevant links are depicted.) **Red dashed link:** Incorrect propagation or incorrectly labeled propagation. **Green dotted link:** Missing propagation.

label	#added	#removed	#sents	#total
acl	14	4	12	68
acl:relcl	13	3	9	190
advcl	32	3	31	167
advmod	46	2	35	6
amod	19	2	14	7
ccomp	9	10	11	50
nmod	32	0	23	102
nmod:poss	11	0	10	18
nsubj	160	30	118	249
nsubj:pass	8	18	25	225
nsubj:xsubj	22	10	17	1688
obj	9	5	12	71
obl	72	0	51	150
ref	12	1	8	61
xcomp	7	5	10	8
all	466	93	386	3060

Table 3: Statistics of modifications made to 1,399 sentences of the EWT. **#sents** reports the number of sentences in which the respective reported changes were made, **#total** reports the number of occurrences of the label in the enhanced layer of the original treebank.

eight kilometres along the Soper River," *nsubj* links should be present in the enhanced layer between both conjuncts of the subject noun phrase and both verbs. However, in the original treebank, the second subject conjunct was never propagated to the second verb (see Figure 2c). Similarly, we also added many relations in cases of **nested coordinations** as in "These Shiite movements had been suppressed by Saddam Hussein's regime, but have now organized and armed themselves." The second conjunct of the conjoined verb phrase is a conjoined verb phrase itself, but the *nsubj* link to "armed" was missing. In total, 194 sentences contain several coordinations, and we modified 92 of them. This phenomenon also accounts for 45 of the added *nsubj* links.

Some originally missing propagations concern **adjectival and adverbial modifiers** (*acl, amod, advcl, advmod*), which are known to be ambiguous cases. In "Handwritten notes and files on a laptop were seized," the adjective "handwritten" clearly modifies the first conjunct "notes" only, but in "Sev-

eral Indian scholars and politicians have been ready to say and endorse anything," the propagation of "several" and "Indian" was added during our modifications. These cases involve world knowledge that the converter currently does not handle.

Finally, consider the sentence "We recognize that the state may not require religious groups to officiate at, or bless, same-gender marriages." Both conjuncts take "marriages" as their argument, but as an *obl* and as an *obj* relation, respectively. The resolution of such non-parallel constructions requires detailed subcategorization information.

4 Modeling

In this section, we describe three approaches to generating links propagated due to coordination: (1) an improved version of an existing converter (Sec. 4.1); (2) ML-based propagation classification operating on basic trees (Sec. 4.2); and (3) a graph-parser based approach for directly predicting edges between tokens (Sec. 4.3). While (1) and (2) may be used to construct "silver standard" enhanced UD graphs from gold trees, (3) is applicable in the automatic parsing setting only.

4.1 Modifications to Rule-based Converter

Based on the error analysis in Sec. 3.3, we modify the rule-based converter by Schuster and Manning (2016) as follows. In order to fix errors related to subject propagation in passive and imperative constructions, we take the conjunction dependent's morphological features into account. In the gold standard, the *Voice* feature is considered to be *active* by default. Hence, if the conjunction dependent does not have a *Voice* feature or is explicitly marked as *active*, an *nsubj:pass* dependency will be propagated as *nsubj*. Similarly, if it has the feature *Mood=Imp*, an *nsubj* link will not be propagated. Our second modification propagates common adjuncts of verbs as well (*obl*, *advmod*, and *advcl*). We maintain the rule from object propagation that a dependency is only propagated if the dependent comes after the potential target in the sentence. Finally, to handle multiple and nested coordinations, we iterate the converter's conjunction propagation function until the dependency graph does not change any more. This allows dependencies that result from propagation to be propagated themselves, retrieving links that would otherwise be missed.

4.2 Conjunction Propagation Classifiers

The core idea of ML-based conjunction propagation classifiers is to take a basic-layer tree and to decide for each incoming or outgoing dependency of the head of a coordinated phrase whether to propagate this dependency to the other coordinated item(s). We refer to the coordinated nodes as **conjunction head** and **conjunction dependent** and to the candidate governor/dependent of the second conjunct as the propagation **target**. In Figure 1, these three nodes correspond to "wrote," "published" and "1954" (or "PEREZ"/"book"), respectively. The output is a binary decision whether to propagate the given dependency or not. In addition to the features described below, we always provide the candidate dependency label and direction.

SVM-based Classifier. We re-implement the method proposed by Nyblom et al. (2013) using scikit-learn's SVC with a polynomial kernel of degree 2.[8] The features comprise morphological information about the tokens for the conjunction head/dependent and the target, as well as structural tree features extracted from the basic-layer tree. For a detailed description, see Appendix D.

Neural network classifier. We pass the sentence through the transformer-based neural language model RoBERTa (Liu et al., 2019) and extract the word embeddings for the first wordpiece tokens of the conjunction head, the conjunction dependent, and the propagation target. In addition, we use equivalents of the SVM tree features using learned embeddings or one-hot encodings (see Appendix D). The inputs are concatenated and fed to a multi-layer perceptron, which then outputs the binary decision whether to propagate the dependency or not. The multi-layer perceptron consists of two linear layers with hidden sizes 1500 and 500 respectively. We implement the model using Huggingface's Transformers library (Wolf et al., 2019). RoBERTa weights are not fine-tuned.

4.3 Graph-Parser Based Edge Prediction

In addition to the above approaches, we also evaluate a graph-parser based approach that predicts dependencies between tokens directly, i.e., which does not rely on a basic-layer tree. Our unfactorized architecture is similar to that of Grünewald and Friedrich (2020), i.e., our model predicts presence of edges and the corresponding labels in a single

[8] https://scikit-learn.org

step, treating nonexistence of an edge as simply another label (∅). As we focus on the dependencies involved in conjunctions, we do not require the parser's output to constitute valid graphs.

Embeddings for input tokens are generated by feeding gold tokens to the RoBERTa tokenizer and then running the resulting word-pieces through the RoBERTa-large model. We then generate an embedding $\mathbf{r}_i$ for the token at position i by forming a weighted sum of the hidden layers' embeddings at the positions corresponding to the first word-piece token of the original token as suggested by Kondratyuk and Straka (2019). Weights for this scalar mixture of layers are learned during training. Layers are randomly dropped during training to prevent the model from focusing on only a single layer.

For each input embedding $\mathbf{r}_i$, we create a head representation h_i^{head} and a dependent representation h_i^{dep} via two feed-forward neural networks:

$$\mathbf{h}_i^{head} = \text{FNN}^{head}(\mathbf{r}_i) \quad (1)$$

$$\mathbf{h}_i^{dep} = \text{FNN}^{dep}(\mathbf{r}_i) \quad (2)$$

For each ordered pair (i, j) of tokens, we feed their respective head and dependent representations to a biaffine classifier (Dozat and Manning, 2017) predicting logits $\mathbf{s}_{i,j}$ over the possible dependency labels. We use these logits to extract the probabilities $P(y_{i,j})$ for each label:

$$\text{Biaff}(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^\top \mathbf{U}\mathbf{x}_2 + W(\mathbf{x}_1 \oplus \mathbf{x}_2) + \mathbf{b} \quad (3)$$

$$\mathbf{s}_{i,j} = \text{Biaff}\big(\mathbf{h}_i^{head}, \mathbf{h}_j^{dep}\big) \quad (4)$$

$$P(y_{i,j}) = \text{softmax}(\mathbf{s}_{i,j}) \quad (5)$$

$\mathbf{U}$, W and $\mathbf{b}$ in (3) are learned parameters; $\oplus$ denotes concatenation. The model is trained to minimize cross entropy loss w. r. t. the true dependency label between each pair of tokens. If a token is not assigned any head due to ∅ scoring highest for all other tokens, we assign the highest-scoring non-∅-relation and the corresponding head.

The model is simply trained to predict all link types in enhanced UD graphs. In the training section of the EWT corpus, we replace every sentence that contains a coordinated verb phrase with our manually corrected version of that sentence, or remove it from the corpus if it is one of the 927 conjunction sentences in the training section which we did not correct. For hyperparameter settings, see Appendix E.

5 Experiments

In this section, we describe our experiments on creating enhanced UD representations for coordinate constructions. Analogous to Nyblom et al. (2013), we measure precision, recall and F1 on enhanced links that are the result of propagation in coordinate constructions. For all experiments, we use gold sentence segmentation and tokenization, and evaluate on our manually corrected sentences from the dev and test sets of the EWT corpus.

5.1 Gold Standard Treebank Enhancing

We first address the research question of how to best generate enhanced representations for treebanks with gold standard basic annotations. We compare the following models: (1) an "Always" baseline, which simply propagates all incoming and outgoing links from the conjunction head to the conjunction dependent(s); (2) the rule-based converter by Schuster and Manning (2016) and the variations thereof we developed inspired by our corpus study; (3) our re-implementation of the SVM-based classifier by Nyblom et al. (2013); and (4) our neural-network (NN) based classifier. The latter uses AdamW (Loshchilov and Hutter, 2017) with a learning rate of 5e-5, a batch size of 1 and early stopping. Table 4 reports the results on the development and test sets of our manually verified conjunction dataset. The recall of the "Always" baseline is not at 100% because a small number of relations change their label during propagation, e.g., *nsubj→nsubj:pass*.

Rule-based conversion. We show results for successively adding components to the original converter (**RBC**). On the test set, adding propagation of non-core dependents and allowing several iterations increases recall and improves F1 by more than 2 points. On the dev set, in contrast, we do not observe these effects.[9] Adding our suggested passive/imperative fix surprisingly decreased performance. Analysis showed that the cases that our converter got wrong were caused by erroneous morphological feature annotations in the basic layer. In sum, our suggested improvements (**RBC2**) of heuristically propagating adjuncts (*obl*, *advmod*, *acl*) and allowing several resolution passes of the

[9] We assume that the reason is the presence of several informal-language sentences in the dev set that include multiple conjunctions (e.g., "etc. etc. etc."), whose annotation is unclear even in the basic gold standard.

converter seem to improve treebank enhancing, provided that the basic layer is correct.

ML-based conversion. Overall, the SVM and NN models show similar performance. As they perform already close to human agreement (see Table 2), further improvement may actually indicate overfitting. On the test set, the ML-based methods outperform the heuristic rule-based methods, surpassing the original converter by over 4 points F1. We conclude that learning structural rules based on actual gold standard data is more effective than hand-designing them. Differences on the dev set are less pronounced despite models being optimized on this data, again hinting to some qualitative differences between the two sets.

In order to determine which sources of information are most relevant, we perform ablation experiments for both classifiers. The features representing the candidate dependency label and the direction of the link are essential and kept in each case. Both the SVM and the NN classifiers draw most of their information from tree-based features. This effect is particularly pronounced for the SVM classifier, where performance drops by 10 to almost 20 points F1 when ommitting these features. The NN classifier's performance does not deteriorate as strongly under the same condition, indicating that some syntactic information can also be retrieved from contextualized word embeddings (see e.g., Tenney et al., 2019). Nonetheless, in most experiments, adding token features improves performance slightly, showing that they do contain important information for propagation decisions.

	Dev			Test		
	P	R	F	P	R	F
"Always" baseline	*23.1*	*99.6*	*37.5*	*28.0*	*99.6*	*43.7*
RBC	**94.8**	86.4	**90.4**	95.2	76.9	85.0
+ non-core deps	93.7	86.4	89.9	94.9	79.7	86.7
+ iteration (**RBC2**)	90.1	86.8	88.4	93.9	81.5	87.2
+ passive fix	91.7	85.3	88.4	**95.7**	78.6	86.3
SVM	87.6	87.9	87.8	93.4	85.4	**89.2**
- tree features	75.5	78.0	76.8	76.5	63.7	69.5
- token features	86.3	87.5	86.9	92.3	85.1	88.5
NN	87.0	87.9	87.4	92.0	**85.8**	88.8
- tree features	87.1	86.4	86.8	88.0	78.6	83.1
- token features	87.3	**88.3**	87.8	92.2	84.3	88.1

Table 4: **Predicting relation propagation for coordinate constructions on gold basic trees**. Precision, recall and F1 on propagated relations in the predicted vs. gold dependency graphs in our manually verified conjunction dataset. The gold dev and test sets contain 273 and 281 instances, respectively.

5.2 Propagating Conjunction Links in Automatic Parsing Setting

For the scenario of parsing from raw tokens, we compare two state-of-the-art parsers, StanfordNLP (Qi et al., 2018) and UDify (Kondratyuk and Straka, 2019), combined with the rule-based converter or ML-based conjunction propagators, and our graph-parser based edge predictor. The latter is trained on the subset of training sentences that either do not contain coordinated verb phrases or that were corrected by us. Hyperparameters and training settings are given in Appendix E.

Results for these experiments can be found in Table 5. The impact of the quality of the parsed basic dependencies is evident: Results are much better for the UDify parser (LAS F1 of 89.4 for basic dependencies on the EWT dev set) than for StanfordNLP (LAS F1 of 87.4). In the automatic setting, our heuristic extensions improve results compared to using the original converter, and there is no decrease in F1 on dev. As in the gold standard settings, ML-based extensions improve upon RBC on test, but not dev. Of the systems based on basic-layer tree parsers, RBC2 works best. However, performance of all pipeline systems show rather poor performance at or below an F1 of 70. Our graph-parser based edge predictor achieves by far the best results, outperforming all other models by a margin of over 7 points F1. This shows that in an automatic setting, most robust results are achieved by directly inducing dependency links between tokens, modeling conjunction only indirectly.

To estimate the impact of our corrections to the gold standard, we also train the graph parser on uncorrected data. The model trained on the corrected data has higher recall, but lower precision. This is expected to some extent as we introduce semantically motivated propagations of adjuncts, and we suspect that they may require a larger training set.

5.3 Discussion

The main insights comparing our experiments in the gold standard vs. the automatic parsing setting are as follows. Overall, our heuristic extensions for the rule-based converter are beneficial in both settings. In the gold setting, ML-based extensions lead to higher accuracy; when applying them on noisy parser output, they do not work well. However, using *one* end-to-end machine-learning model

	Dev			Test		
	P	R	F	P	R	F
Stanford+RBC	70.8	63.0	66.7	56.5	47.7	51.7
Stanford+RBC2	68.7	65.2	66.9	56.2	50.2	53.0
Stanford+SVM	64.3	65.2	64.7	54.7	49.8	52.1
Stanford+NN	64.3	65.2	64.7	54.4	50.2	52.2
UDify+RBC	72.8	67.8	70.2	71.8	58.0	64.2
UDify+RBC2	71.9	68.5	70.2	75.0	61.9	67.8
UDify+SVM	70.6	68.5	69.5	70.9	59.1	64.5
UDify+NN	69.9	68.9	69.4	70.4	60.1	64.9
GBP (orig. data)	**83.1**	74.0	78.3	**86.1**	66.2	74.8
GBP (our data)	82.3	**75.1**	**78.5**	82.5	**68.7**	**75.0**

Table 5: **Predicting relation propagation for coordinate constructions on parser output**. Otherwise same evaluation setup as in Table 4.

directly to generate enhanced representations for conjunctions outperforms the pipeline version. A possible reason for this might be that these models were all developed on gold data, while the graph-based parser does not rely on potentially wrong structural tree features and is also able to use internal confidence information for edges. Another advantage of the end-to-end model may stem from the fact that its training allows to leverage semantic information from training data of a larger number of dependency links, i.e., including those not occurring in coordinate constructions. This points to a promising future research direction, i.e, generating additional semi-artificial training data for conjunction propagation.

6 Conclusion and Outlook

We have presented a large-scale manually curated **dataset for conjunction propagation** in English. In contrast to previous work focusing on high-precision rule-based propagation, we propagate links in all cases that semantically suggest argument or adjunct sharing. In the gold standard treebank enhancing setting, we found **ML-based models** to outperform the de-facto standard rule-based converter by learning to exploit mostly structural features. However, one of our main insights is that neither rule-based nor ML-based classifiers work well on noisy parser output precisely because of this reliance on structural information. We propose to use a graph-parser based edge predictor instead and show that it outperforms pipeline-based models by a large margin. Our model reaches F1 scores between 0.75 and 0.78 with a precision of more than 0.82, a level of performance that may already be useful in downstream tasks.

Our models could be used for creating high-quality enhanced-level representations of conjunctions for the remaining English data, and could thus help in a UD community effort to continuously **improve the UD treebanks**. Future work also includes the study of conjunction propagation methods for **further languages**. Our in-depth study on English data provides several insights that we expect to be transferable cross-linguistically. First, conjunction propagation can to some extent be addressed using heuristic rules, but capturing the full semantic nature of the task requires manual annotation. Second, given appropriate training data, our machine-learning based approaches are also applicable to other languages.

In addition, it would be interesting to see if manually annotated data for coordinate constructions may be useful in natural language understanding tasks such as natural language inference (NLI). This is especially true for "stress test" datasets such as CONJNLI (Saha et al., 2020), which are designed to specifically test models' capabilities to process coordination.

Finally, as morphological features are generally important for this task, improving their automatic prediction (see e.g., Ramm et al., 2017; Myers and Palmer, 2019) as well as UD's gold standard seems to be a promising way to go. Our work has demonstrated the value of a linguistically motivated corpus study of a syntactic-semantic phenomenon, and shown that given manually curated data, rules for conjunction propagation can be learned effectively.

Acknowledgments

We would like to thank Sherry Tan and Johannes Hingerl for their support, as well as Jonas Kuhn, Heike Adel, Jannik Strötgen, and the anonymous reviewers for their useful comments regarding this work. In addition, we are grateful to the UD community, especially Nathan Schneider, for answering our questions.

References

Chiara Alzetta, Felice Dell'Orletta, Simonetta Montemagni, Maria Simi, and Giulia Venturi. 2018. Assessing the impact of incremental error detection and correction. a case study on the Italian universal dependency treebank. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 1–7, Brussels, Belgium. Association for Computational Linguistics.

Gosse Bouma, Djamé Seddah, and Daniel Zeman. 2020. Overview of the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, Seattle, US. Association for Computational Linguistics.

Mathieu Dehouck, Mark Anderson, and Carlos Gómez-Rodríguez. 2020. Efficient EUD parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 192–205, Online. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Stefan Grünewald and Annemarie Friedrich. 2020. RobertNLP at the IWPT 2020 shared task: Surprisingly simple enhanced UD parsing for English. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 245–252, Online. Association for Computational Linguistics.

Han He and Jinho D. Choi. 2020. Adaptation of multilingual transformer encoder for robust enhanced Universal Dependency parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 181–191, Online. Association for Computational Linguistics.

Johannes Heinecke. 2019. ConlluEditor: a fully graphical editor for universal dependencies treebank files. In *Proceedings of the Third Workshop on Universal Dependencies (UDW, SyntaxFest 2019)*, pages 87–93, Paris, France. Association for Computational Linguistics.

Johannes Heinecke. 2020. Hybrid enhanced Universal Dependencies parsing. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 174–180, Online. Association for Computational Linguistics.

Daniel Hershcovich, Miryam de Lhoneux, Artur Kulmizev, Elham Pejhan, and Joakim Nivre. 2020. Køpsala: Transition-based graph parsing via efficient training and effective encoding. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 236–244, Online. Association for Computational Linguistics.

Jenna Kanerva, Filip Ginter, and Sampo Pyysalo. 2020. Turku enhanced parser pipeline: From raw text to enhanced graphs in the IWPT 2020 shared task. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 162–173, Online. Association for Computational Linguistics.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. 2014. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4585–4592, Reykjavik, Iceland. European Language Resources Association (ELRA).

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy. European Language Resources Association (ELRA).

Jonathan May and Jay Priyadarshi. 2017. SemEval-2017 task 9: Abstract Meaning Representation parsing and generation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 536–545, Vancouver, Canada. Association for Computational Linguistics.

Skatje Myers and Martha Palmer. 2019. ClearTAC: Verb tense, aspect, and form classification using

neural nets. In *Proceedings of the First International Workshop on Designing Meaning Representations*, pages 136–140, Florence, Italy. Association for Computational Linguistics.

Joakim Nivre, Paola Marongiu, Filip Ginter, Jenna Kanerva, Simonetta Montemagni, Sebastian Schuster, and Maria Simi. 2018. Enhancing universal dependency treebanks: A case study. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 102–107, Brussels, Belgium. Association for Computational Linguistics.

Jenna Nyblom, Samuel Kohonen, Katri Haverinen, Tapio Salakoski, and Filip Ginter. 2013. Predicting conjunct propagation and other extended Stanford dependencies. In *Proceedings of the Second International Conference on Dependency Linguistics (DepLing 2013)*, pages 252–261, Prague, Czech Republic. Charles University in Prague, Matfyzpress, Prague, Czech Republic.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, and Zdeňka Urešová. 2015. SemEval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 915–926, Denver, Colorado. Association for Computational Linguistics.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Yi Zhang. 2014. SemEval 2014 task 8: Broad-coverage semantic dependency parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 63–72, Dublin, Ireland. Association for Computational Linguistics.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, Brussels, Belgium. Association for Computational Linguistics.

Anita Ramm, Sharid Loáiciga, Annemarie Friedrich, and Alexander Fraser. 2017. Annotating tense, mood and voice for English, French and German. In *Proceedings of ACL 2017, System Demonstrations*, pages 1–6, Vancouver, Canada. Association for Computational Linguistics.

Swarnadeep Saha, Yixin Nie, and Mohit Bansal. 2020. ConjNLI: Natural language inference over conjunctive sentences. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8240–8252, Online. Association for Computational Linguistics.

Sebastian Schuster, Éric Villemonte de La Clergerie, Marie Candito, Benoît Sagot, Christopher Manning, and Djamé Seddah. 2017. Paris and Stanford at EPE 2017: Downstream evaluation of graph-based dependency representations. In *Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation (EPE 2017)*, pages 47–59.

Sebastian Schuster and Christopher D. Manning. 2016. Enhanced English universal dependencies: An improved representation for natural language understanding tasks. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 2371–2378, Portorož, Slovenia. European Language Resources Association (ELRA).

Sebastian Schuster, Joakim Nivre, and Christopher D. Manning. 2018. Sentences with gapping: Parsing and reconstructing elided predicates. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1156–1168, New Orleans, Louisiana. Association for Computational Linguistics.

Maria Simi and Simonetta Montemagni. 2018. Bootstrapping enhanced universal dependencies for italian. In *5th Italian Conference on Computational Linguistics, CLiC-it 2018*, volume 2253. CEUR-WS.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Xinyu Wang, Yong Jiang, and Kewei Tu. 2020. Enhanced Universal Dependency parsing with second-order inference and mixture of training data. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, pages 215–220, Online. Association for Computational Linguistics.

Guillaume Wisniewski. 2018. Errator: a tool to help detect annotation errors in the universal dependencies project. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Appendix

A Detailed converter description

In this section, we describe the algorithm implemented by the converter proposed by Schuster and Manning (2016).[10]

For each *conj* relation, the converter decides whether links ending or starting at the conjunction head (`gov`) should be propagated to the conjunction dependent (`dep`):

1. Governors of `gov` are always propagated to `dep`, unless the relation is explicitly treated as an exception (e.g., *vocative*, *discourse*, or *root*).

2. Dependents of `gov` are propagated to `dep` as follows:

   (a) If the dependent is attached via *nsubj* or *csubj*, it is only propagated if `dep` does not already have a subject. If `dep` has an *aux:pass* dependent, the relation is propagated as *nsubj:pass* / *csubj:pass*.

   (b) If the dependent is attached via a non-subject core relation (*obj*, *iobj*, *ccomp*, or *xcomp*), it is propagated if and only if it comes after `dep` in the linear order of the sentence.

   (c) Non-core dependents (such as *obl*) are never propagated.

The algorithm is able to handle many syntactically ambiguous cases, provided the underlying basic dependencies have resolved the ambiguity correctly. Consider the sentence "She was reading or watching a movie." If "movie" is correctly attached as an object of the conjunction dependent "watching," it will not be propagated to "reading" in the enhanced representation.

B Inter-annotator agreement study

Detailed comparisons between the three annotators can be found in Table 6, Table 7, and Table 8. In our study, we consider only links that are part of the enhanced layer, but not of the basic layer. For each annotator, we count for each label how often it occurs as an incoming or outgoing relation of a conjunct (columns labeled with the annotator's ID). Formally, the set E^l_A is the set of enhanced-layer edges that are (i) not present in the basic layer and (ii) involved in conjunctions as incoming or outgoing links of the conjuncts, with label l marked by annotator A. We also count the overlap of links for pairs of annotators. Using these counts, we then compute precision, recall and F1, treating one annotator as the system and one as the gold standard. For instance, when treating A as the gold standard and B as the system, this leads to:

$$Precision_{BA} = \frac{|E^l_A \cap E^l_B|}{E^l_B} \quad (6)$$

$$Recall_{BA} = \frac{|E^l_A \cap E^l_B|}{E^l_A} \quad (7)$$

Note that when reversing this order, P and R are simply reversed, F1 stays the same.

The following numbers compare each annotator to the original gold standard (not in tables). For modifier clauses (*acl*, *advcl*) and adverbials (*advmod*), B was the most aggressive in propagating dependencies, adding 55 links in total for these labels, while A and C only added 39 and 32 links, respectively. While all annotators propagated *obl* dependencies roughly to the same extent, agreement was high between A and B but lower (F1 64-68%) between C and the others, indicating that there are more ambiguities among these dependencies as well. Annotator C is generally more conservative in propagating dependencies. This is reflected in the relatively low recall when comparing to the other annotators, as well as the lower overall number of added links (285 as compared to 309 for A and 312 for B).

[10] https://github.com/stanfordnlp/CoreNLP/blob/master/src/edu/stanford/nlp/trees/ud/UniversalEnhancer.java

	B	A	A&B	P	R	F1
acl	7	7	7	100.0	100.0	100.0
acl:relcl	12	8	8	66.7	100.0	80.0
advcl	24	17	17	70.8	100.0	82.9
advmod	10	5	4	40.0	80.0	53.3
amod	4	6	4	100.0	66.7	80.0
ccomp	12	13	12	100.0	92.3	96.0
compound	3	3	3	100.0	100.0	100.0
csubj	2	2	2	100.0	100.0	100.0
nmod	8	8	8	100.0	100.0	100.0
nsubj	62	66	62	100.0	93.9	96.9
nsubj:pass	5	5	5	100.0	100.0	100.0
nsubj:xsubj	6	7	6	100.0	85.7	92.3
obj	26	25	25	96.2	100.0	98.0
obl	29	26	25	86.2	96.2	90.9
ref	5	5	5	100.0	100.0	100.0
xcomp	7	7	7	100.0	100.0	100.0
total	222	210	200	90.1	95.2	92.6

Table 6: **Agreement** of Annotator A vs. Annotator B on links involved in coordinate constructions in the enhanced layer. For P/R computation, A was treated as the gold standard and B as the system.

	C	A	A&C	P	R	F1
acl	7	7	7	100.0	100.0	100.0
acl:relcl	4	8	4	100.0	50.0	66.7
advcl	11	17	10	90.9	58.8	71.4
advmod	4	5	0	0.0	0.0	0.0
amod	4	6	4	100.0	66.7	80.0
ccomp	11	13	11	100.0	84.6	91.7
compound	3	3	3	100.0	100.0	100.0
csubj	2	2	2	100.0	100.0	100.0
mark	2	0	0	0.0	0.0	0.0
nmod	6	8	6	100.0	75.0	85.7
nsubj	61	66	61	100.0	92.4	96.1
nsubj:pass	5	5	5	100.0	100.0	100.0
nsubj:xsubj	6	7	6	100.0	85.7	92.3
obj	25	25	23	92.0	92.0	92.0
obl	18	26	18	100.0	69.2	81.8
ref	2	5	2	100.0	40.0	57.1
xcomp	7	7	7	100.0	100.0	100.0
total	178	210	169	94.9	80.5	87.1

Table 7: **Agreement** of Annotator A vs. Annotator C on links involved in coordinate constructions in the enhanced layer. For P/R computation, A was treated as the gold standard and C as the system.

	C	B	B&C	P	R	F1
acl	7	7	7	100.0	100.0	100.0
acl:relcl	4	12	4	100.0	33.3	50.0
advcl	11	24	11	100.0	45.8	62.9
advmod	4	10	3	75.0	30.0	42.9
amod	4	4	4	100.0	100.0	100.0
ccomp	11	12	10	90.9	83.3	87.0
compound	3	3	3	100.0	100.0	100.0
csubj	2	2	2	100.0	100.0	100.0
mark	2	0	0	0.0	0.0	0.0
nmod	6	8	6	100.0	75.0	85.7
nsubj	61	62	61	100.0	98.4	99.2
nsubj:pass	5	5	5	100.0	100.0	100.0
nsubj:xsubj	6	6	6	100.0	100.0	100.0
obj	25	26	24	96.0	92.3	94.1
obl	18	29	18	100.0	62.1	76.6
ref	2	5	2	100.0	40.0	57.1
xcomp	7	7	7	100.0	100.0	100.0
total	178	222	173	97.2	77.9	86.5

Table 8: **Agreement** of Annotator B vs. Annotator C on links involved in coordinate constructions in the enhanced layer. For P/R computation, B was treated as the gold standard and C as the system.

C Statistics on Treebank Modifications

As mentioned in Sec. 3.3, we made changes to the original treebank in 1,417 sentences containing coordinate verb phrases. While we focused on the dependency links starting or ending at conjuncts, we also fixed some additional errors that we spotted during this process. Table 9 gives statistics on these modifications (compare to Table 3, which includes only changes related to conjuncts).

label	#added	#removed	#sents	#total
acl	18	5	14	749
acl:relcl	15	5	9	5,086
advcl	39	11	36	2,055
advmod	46	4	37	4,426
amod	20	3	15	3,570
case	2	1	2	6,415
ccomp	12	16	13	1,003
det	2	1	2	6,448
nmod	32	2	24	3,279
nmod:poss	12	1	10	7,485
nsubj	194	41	144	7,815
nsubj:pass	9	21	25	1,381
nsubj:xsubj	35	20	31	8,758
nummod	3	0	3	814
obj	10	10	14	4,673
obl	78	5	56	5,478
punct	3	3	3	9,508
ref	21	2	16	348
xcomp	14	6	12	758
all	565	157	466	80049

Table 9: Statistics of modifications made to 1,417 sentences of the EWT, including both basic and enhanced layer. **#sents** reports the number of sentences in which the respective reported changes were made, **#total** reports the number of occurrences of the label in the original treebank.

D ML-based classifiers: features

Table 10 lists the features used in our SVM and NN models. Token features are extracted for conjunction head, conjunction dependent, and propagation target each. In addition to the listed features, we also experimented with including lemmas and POS tags, but did not find them to be useful in our ablation experiments.

E Graph-based edge predictor: Training Setup

Label lexicalization. At training time, we only use a limited label set of 56 labels where lexical material is replaced with placeholders, such as *obl:[case]*. At prediction time, we retrieve the missing lexical material from the dependency graph in a rule-based fashion. In the simplest case, this means simply substituting the word form of the dependent of the required type (e.g., a *case* relation). In conjunctions, the token in question may not have its own dependent of the correct type, instead "inheriting" if from its conjunction head. In that case, we retrieve the lexical material from the conjunction head's dependent.

Hyperparameters We perform only a minimal amount of hyperparameter tuning, mostly sticking with the values used by Kondratyuk and Straka (2019). One notable exception is the training regime, where we found low batch size and the AdamW optimizer to yield the best results. The full hyperparameter configuration can be found in Table 11.

RoBERTa embeddings	
Embeddings dimension	1024
Token mask probability	0.15
Layer dropout	0.1
Hidden dropout	0.2
Attention dropout	0.2
Output dropout	0.5
Biaffine classifier	
Hidden size	1024
Dropout	0.33
AdamW Optimizer	
Batch size	5
Learning rate	$5e^{-6}$
β_1, β_2	0.9, 0.999
Weight decay	0.0

Table 11: Hyperparameters for our graph-based parser.

Feature name	Description	SVM	NN
Instance features			
dependency label	label of candidate link	one-hot	50-dim. embedding
incoming/outgoing	whether the dependency being propagated is an outgoing or incoming link at the conjunction head	one-hot	50-dim. embedding
midrule ***Token features***			
morphological features	values of the *Number*, *Person*, *VerbForm*, and *Voice* features	one-hot	-
contextualized word embeddings	word embeddings as generated by the RoBERTa-base model	-	768-dim. embedding
Tree features			
linear dependency direction	whether the linear direction of the candidate dependency is the same as for the dependency being propagated (both-left, both-right, or differing-directions)	one-hot	50-dim. embedding
existing dependency	whether the conjunction dependent already has a dependency of this type (only relevant for outgoing links)	one-hot	50-dim. embedding
outgoing dependencies (head)	set of outgoing dependencies of the conjunction head	one-hot	one-hot
outgoing dependencies (dep)	set of outgoing dependencies of the conjunction dependent	one-hot	one-hot
# coord.-items	number of items in the coordination	one-hot	scalar

Table 10: Description of feature sets used in ML-based conjunction propagation models.