Enriching ASR Lattices with POS Tags for Dependency Parsing

Moritz Stiefel and **Ngoc Thang Vu**
Institute for Natural Language Processing (IMS)
Universität Stuttgart
Pfaffenwaldring 5B
70569 Stuttgart
{moritz.stiefel,thang.vu}@ims.uni-stuttgart.de

Abstract

Parsing speech requires a richer representation than 1-best or n-best hypotheses, e.g. lattices. Moreover, previous work shows that part-of-speech (POS) tags are a valuable resource for parsing. In this paper, we therefore explore a joint modeling approach of automatic speech recognition (ASR) and POS tagging to enrich ASR word lattices. To that end, we manipulate the ASR process from the pronouncing dictionary onward to use word-POS pairs instead of words. We evaluate ASR, POS tagging and dependency parsing (DP) performance demonstrating a successful lattice-based integration of ASR and POS tagging.

1 Introduction

Parsing speech is an essential part (Chow and Roukos, 1989; Moore et al., 1989; Su et al., 1992; Chappelier et al., 1999; Collins et al., 2004) of spoken language understanding (SLU) and difficult because spontaneous speech and syntax clash (Ehrlich and Hanrieder, 1996; Charniak and Johnson, 2001; Béchet et al., 2014). Pipeline approaches concatenating a speech recognizer, a POS tagger and a parser often rely on n-best hypotheses decoded from lattices. While n-best hypotheses cover more of the hypothesis space than the 1-best hypothesis, they are redundant and incomplete. Lattices on the other hand are efficiently representing all hypotheses under consideration and therefore allow recovery from more ASR errors. Recent work on recurrent neural network architectures with lattices as input (Ladhak et al., 2016; Su et al., 2017) promises the use of enriched lattices in SLU.

The main contribution of this work is establishing a joint ASR and POS tagging approach using the Kaldi (Povey et al., 2011) toolkit. To that end, we enrich the ASR word lattices with POS labels for all possible hypotheses on the word level. This enables subsequent natural language processing (NLP) machinery to use these syntactically richer lattices. We present our proposed method in detail including Kaldi specifics and address problems that occur when data that requires both speech and text information is used. Our results show a slight but consistent improvement of the joint model throughout the evaluations in ASR, POS tagging and DP performance.

2 Resources

We need a data resource with rich annotations for training our integrated model. Since the training process requires audio transcriptions, POS labels and gold-standard syntax annotations, all of these need to be available. Considering the general premise in data-driven methods that more data is better data, we choose the Switchboard-1 Release 2[1] (Godfrey et al., 1992) corpus with about 2400 dialogs. The Switchboard (SWBD) corpus has more recently been furnished with the NXT Switchboard annotations[2] (Calhoun et al., 2010). NXT provides a plethora of annotations and most importantly for our work, an alignment of Treebank-3[3] (Marcus et al., 1999) text and SWBD transcriptions[4]. While the Treebank-3 corpus pro-

[1] LDC: https://catalog.ldc.upenn.edu/LDC97S62 (Godfrey and Holliman, 1993)
[2] LDC (under CC): https://catalog.ldc.upenn.edu/LDC2009T26 (Calhoun et al., 2009)
[3] Treebank-3 at the LDC: https://catalog.ldc.upenn.edu/LDC99T42
[4] We used the corrected Mississippi State (MS-State) transcriptions: https://www.isip.piconepress.com/projects/switchboard/

Proceedings of the First Workshop on Speech-Centric Natural Language Processing, pages 37–47
Copenhagen, Denmark, September 7–11, 2017.

vides syntax and POS tags, the transcriptions are timestamped. The alignment of these two resources offered by the NXT corpus contains all necessary annotations.

2.1 Audio

Kaldi's SWBD *s5c* recipe subsets the SWBD (LDC97S62) corpus into various training and development sets for acoustic model (AM) and language model (LM) training. For ASR evaluation, the *s5c* recipe uses a separate evaluation corpus LDC2002S09[5] of previously unreleased SWBD conversations (Linguistic Data Consortium, 2002), which was not available to us. Likewise unavailable were the Fisher corpora LDC2004T19[6] (Cieri et al., 2004) and LDC2005T19[7] (Cieri et al., 2005), which contain transcripts of conversational telephone speech for language modeling. We utilize the available SWBD data (the training set in the *s5c* recipe) and split it into training, development and evaluation set. Our results are therefore not directly comparable to other results generated from the Kaldi *s5c* recipe. We instead split our sets after the Treebank-3 splits as proposed by Charniak and Johnson (2001). This leads to less training data compared to the standard *s5c* recipe, but also yields splits common in parsing. A data summary of our SWBD splits is given in Table 1. The *lmdev* section of the SWBD corpus serves as the LM's development set and was "reserved for future use" (Charniak and Johnson, 2001, p. 121).

Set	Conv. IDs	# utt.	# tok.
train	2xxx-3xxx	90823	677160
dev	4519-4936	5697	50148
eval	4004-4153	5822	48320
lmdev	4154-4483	5949	50017

Table 1: Summary of SWBD data splits. The columns for utterances, tokens, average tokens per utterance and vocabulary depend on the choice of the transcription. These are the counts for our Treebank-3 transcription.

2.2 Transcription

While the NXT annotations provide a link between MS-State transcriptions and Treebank-3 text, we exploit this link only for the MS-State transcription's timestamps and base our lexicon and LMs on the Treebank-3 text, rather than the MS-State transcriptions. This introduces a number of text-audio mismatches, or in other words, what is said is not what is in the annotated text. Figure 1 illustrates contractions as one characteristic difference in the tokenization of the two transcriptions: "doesn't" is represented as two tokens in the Treebank-3 data, while it is expressed as one token in the MS-State version. The second important aspect of choosing the Treebank-3 over the MS-State transcription, is the incongruity of utterances (cf. Calhoun et al., 2010, ch. 3.3, p. 393ff). Training and evaluation become easier if the utterances are congruent in the transcription and the Treebank-3 data with the syntactical parses. We decided to directly base the transcriptions on these annotations.

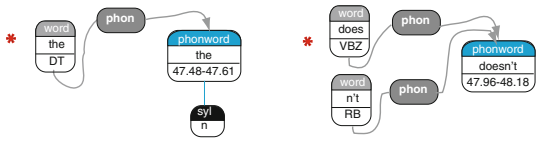

Figure 1: MS-State vs Treebank-3 transcription, from Calhoun et al. (2010, p. 392). Treebank-3 transcriptions (*word*, in light gray) are mapped to the MS-State transcriptions (*phonword* in blue) through 1-to-n relations, where multiple words in one transcription can be linked to one in the other. The box colored in black with *syl/n* in it depicts a unstressed syllable of a different annotation layer we do not consider here.

2.3 Syntax annotation

The linguistic structure annotated in the SWBD Treebank-3 section is available through the NXT Switchboard annotations and is based on the Treebank-3 text. Choosing the Treebank-3 transcription as the gold standard for the ASR system directly yields Treebank-style tokens in the recognized speech. The POS tagset (Calhoun et al., 2010, p. 394) consists of the 35 POS tags[8] in the Treebank-3 tagset. Disfluencies in the SWBD corpus are annotated following Shriberg (1994) and they are present in the Treebank-3 annotations.

3 Proposed method

First, we describe the ASR component based on the default Kaldi *s5c* recipe that generates POS-enriched word lattices in detail. Second, we introduce the POS taggers considered for the pipeline system. Third, we briefly characterize the dependency parser in our experiments.

[5] https://catalog.ldc.upenn.edu/LDC2002S09
[6] https://catalog.ldc.upenn.edu/LDC2004T19
[7] https://catalog.ldc.upenn.edu/LDC2005T19

[8] It is the PTB tagset without punctuation (which is covered by SYM and the remaining nine punctuation tags).

3.1 ASR with POS tagging

Starting from the *s5c* recipe, all but the acoustic modeling part underwent significant changes. The pronouncing dictionary (or lexicon), LM and resulting decoding graph now all contain word-POS pairs rather than words. We are going to outline this process step by step.

Corpus setup: Our model does not access resources other than the Switchboard-1 Release 2 (LDC97S62, with updates and corrected speaker information) data, the MS-State transcription and the Switchboard NXT corpus as described in Section 2. All transcription-based resources are being lowercased as they are in the *s5c* recipe scripts.

Transcription generation: To get a Treebank-style transcription, we query the NXT annotation corpus for pointers from MS-State tokens to Treebank-3 tokens. With this mapping, we pick the POS tags for the Treebank-3 orthography and the timestamps for the MS-State words. An example for the POS-tagged gold standard transcription is: "`are|VBP you|PRP ready|JJ now|RB`".

POS-enriched lexicon: We first append the lexicon with some handcrafted lexical additions for contractions of auxiliaries and adjust for tokenization differences between the source MS-State format and the target Treebank-3 format. The pronunciation of the resulting partial words is taken from the respective full entries in the dictionary supplied with the MS-State transcriptions. The lexical unit "won't", for example, is mapped to the pronunciation "w ow n t" in the MS-State version, but is not readily merged from the existing partial words ("wo" and "n't") in the MS-State lexicon and therefore is a lexical addition. Other auxiliaries, like "can't" that needs to be split as "ca n't" to conform with the Treebank-3 tokenization, and partial words in general, are added in the lexicon conversion via automated handling where all partials exist.

For all gold standard occurrences of word-POS combinations, we copy the words' pronunciations for all of the POS tags they occur with. Partial words starting with a hyphen are automatically added to the lexicon without the hyphen to account for tokenization differences. Duplicate word-POS pairs are excluded. Figure 2 shows part of the resulting POS-enriched lexicon, where "read" occurs with four different POS tags and two distinct pronunciations. We use "`<unk>|XX`" for unknown tokens. Note that our scheme can overgenerate word-POS combinations, as it does not check whether the pronunciation variation occurs with all POS tags of a word (compare left and right parts of Figure 2).

```
read|VB r eh d
read|VB r iy d
read|VBD r eh d
read|VBD r iy d
read|VBN r eh d
read|VBN r iy d
read|VBP r eh d
read|VBP r iy d
```

```
read r eh d
read r iy d
```

Figure 2: Pronunciation entries for "read" in the lexicon, with (left) and without (right) POS tags.

Language modeling: LM training is performed on the *train* set with the *lmdev* set as heldout data. We train the LM on the POS-enriched transcription directly. See Figure 3 for example trigrams.

```
-0.000432954    we|PRP ca|MD n't|RB
-0.0004147099   's|BES kind|RB of|RB
-0.0003858729   they|PRP ca|MD n't|RB
-0.0002859116   just|RB kind|RB of|RB
-0.0001056216   you|PRP ca|MD n't|RB
```

Figure 3: Top 5 trigrams in the Joint-LM, based on the conditional log probabilities in the first column.

Different from the *s5c* recipe, we compute trigram and bigram LMs with SRILM[9] (Stolcke, 2002) and "`<unk>|XX`" as unknown token. As discussed in Section 2, we did not use SWBD-external resources for mixing and interpolating our LMs. We use SRILM with modified Kneser-Ney smoothing (Chen and Goodman, 1999) with interpolated estimates, and use only words occurring in the specified vocabulary and not in the count files. We report LM perplexity (PPL) on the *lmdev* held-out data in Table 2. Note that the joint model LM in Table 2 encounters 150 OOV tokens (e.g. hyphenated numerals like "thirty-seven"). The PPLs increase slightly for the joint model because the vocabulary has n entries for each word, where n is the number of POS tags the word occurs with.

Acoustic modeling: We use the original *s5c* recipe and only adjust the training, development and evaluation splits after Charniak and Johnson (2001) (cf. Table 1). None of the other aforementioned adaptations are applied and the manually corrected MS-State transcriptions are in use. The *tri4* model in the *s5c* recipe is a triphone

[9] http://www.speech.sri.com/projects/srilm/

LM	PPL
Baseline 2-gram	89.4
Baseline 3-gram	76.3
Joint 2-gram	96.4
Joint 3-gram	84.2

Table 2: PPL and OOVs on *lmdev*.

(with one context phone to the left and right) model which was trained with speaker-adaptive training (SAT, Anastasakos et al., 1996; Povey et al., 2008) technique using feature-space maximum likelihood linear regression (fMLLR, Gales, 1998). We train this *tri4* AM on the training split in Table 1 with duplicate utterances removed.

3.2 Baseline POS tagging

We perform POS tagging with three out-of-the-box taggers, two of them with pretrained models, and choose the best one for our baseline pipeline model.

NLTK's (Bird et al., 2009) former default maximum entropy-based (ME) POS tagger with the pretrained model trained on WSJ data from the PTB (for an overview, see Taylor et al., 2003) is the first tagger and we term it *ME.pre*. We also train a ME POS tagger[10] that is implemented after Ratnaparkhi (1996) on the first 70,000 sentences[11] of our SWBD training split, described in Section 2, and denote our self-trained model by *ME.70k*. We configure the ME classifier to use the optimized version of MEGAM (Daumé III, 2004) for speed.

The second tagger is NLTK's current default tagger, based on a greedy averaged perceptron (AP) tagger developed by Matthew Honnibal[12]. We name the AP tagger with the pretrained NLTK model *AP.pre*, and the same tagger trained on the full training split *AP*.

To have an NLTK-external industry-standard POS tagger in our comparison, we also run spaCy's POS tagger (see `https://spacy.io/`, we used spaCy in version 1.0.3) with its pretrained English model (also trained with AP learning).

[10] Available in NLTK and at: `https://github.com/arne-cl/nltk-maxent-pos-tagger`

[11] The sentences are sorted by their utterance id. The full training set was not computationally feasible: MEGAM threw an "out of memory" error.

[12] `https://explosion.ai/blog/part-of-speech-pos-tagger-in-python`

3.3 Dependency parsing

In this work, we compare dependency parsing results of (a) the 1-best hypothesis of the baseline *tri4* ASR system with the self-trained AP POS tagger and (b) the 1-best hypotheses of our joint model. We use a greedy neural-based dependency parser reimplemented after the greedy baseline in Weiss et al. (2015).

The parser's training set is the gold standard data of the training split and identical for the *tri4* and the Joint-POS model with 62728 trainable sentences out of 63304 (= 99.09%). In this evaluation, we tune the parser based on development data and use word- and POS-based features. The parser implementation uses averaged stochastic gradient descent proposed independently by Ruppert (1988) and Polyak and Juditsky (1992) with momentum (Rumelhart et al., 1986). We do not embed any external information.

4 Results

Our evaluation includes intermediate ASR and POS tagging results and a DP-based evaluation. We evaluate partially correct ASR hypotheses with a simplistic scoring method that allows imprecise scoring when the recognized sequence of tokens does not match the gold standard.

4.1 ASR

We test our joint ASR and POS model against the default *tri4* model in a ASR-only evaluation of the 1-best hypotheses. As we generate the word-POS pairs jointly and they are part of the ASR hypotheses, we strip the POS tags for the word-only evaluation in Table 3. We evaluate the ASR step based on word error rate (WER) and sentence error rate (SER).

Set	Default *tri4*	Joint-POS
dev	**28.75** (65.83)	28.93 **(65.28)**
eval	29.41 (64.41)	**29.26 (64.15)**

Table 3: ASR results: numbers are WER (SER) as percentages. POS tags stripped when evaluating joint model.

Recall that these results are not directly comparable to other ASR results on the SWBD corpus, because of our data splits with less training data and use of the Treebank-3 transcription. In the unaltered (apart from the splits, see Section 2.1), original *s5c* recipe, the WER on the

eval set with the original MS-State transcriptions (48926 tokens, 4331 utterances) is 26.51% with a SER of 67.91%. Compared to the baseline, the results of our Joint-POS model are slightly better for the *dev* set and *eval* set in SER, and for the *eval* set also in WER.

4.2 POS tagging

We present an evaluation of our joint model's performance up to the baseline model's POS tagging step. We compare against the POS tagger performance on the 1-best ASR hypotheses in the pipeline approach. As the 1-best hypotheses of joint and pipeline model can differ, we evaluate the POS tagging step on ASR output against the word-POS pair Treebank-3 gold standard by means of WER.

Tagger	dev	eval
ME.pre	43.29 (94.23)	44.49 (94.19)
AP.pre	45.46 (95.84)	46.18 (95.74)
spaCy.pre	39.17 (82.83)	40.42 (81.86)
ME.70k	33.24 (68.18)	36.35 (54.98)
AP	32.30 (67.67)	33.10 (66.85)
Joint-POS	**32.05 (67.32)**	**32.52 (66.52)**

Table 4: POS tagging results: numbers are WER (SER) on the 1-best hypotheses. ME.70k is trained on the first 70,000 training set sentences. A model name ending in *.pre* indicates the use of a pretrained model. Model names without dotted endings are trained on the full SWBD training set. Best scores per set are in boldface.

Table 4 shows that the Joint-POS model consistently outperforms the baseline POS taggers on both sets. The pretrained models clearly have not been trained on speech data and unsurprisingly perform poorly. Our self-trained ME and AP models improve at least 6% in WER and 15% in SER over the pretrained models. The margin by which our joint model surpasses the self-trained AP tagger is small with an improvement of 0.25% WER on the *dev* and 0.58% WER on the *eval* set. The self-trained AP tagger performed best of the baseline taggers and we therefore use it in for the DP-based evaluation in the next section.

4.3 DP

We evaluate our joint ASR-POS model on the target task by running a dependency parser on POS-tagged 1-best hypotheses. In the competing pipeline model, we score the output of the default *tri4* ASR 1-best hypotheses tagged by the AP tagger we trained ourselves. All results in Table 5 and Table 6 show that our joint model does profit from the joint ASR and POS modeling in our approach.

			tri4		Joint-POS	
Set	#utts	#tokens	UAS	LAS	UAS	LAS
dev	900	4881	94.30	92.71	**95.41**	**93.63**
eval	882	4827	94.68	93.06	**94.92**	**93.52**
dev$_P$	942	5261	94.16	92.38	—	—
eval$_P$	921	5134	94.06	92.31	—	—
dev$_J$	932	5158	—	—	94.65	92.88
eval$_J$	921	5137	—	—	94.61	92.93

Table 5: Parsing results for subsets of correct tokenizations. Labeled attachment scores (LAS) and unlabeled attachment scores (UAS) given as percentages. Best scores on the common sets in boldface.

Table 5 features evaluations of six different development and evaluation sets. The sets named *dev* and *eval* are the common subsets of token-level correct hypotheses that the pipeline and joint model share and therefore can be directly compared on. The sets indexed with a P or J are the token-level correct hypotheses for the pipeline and joint model respectively. As the models are not identical with respect to their 1-best hypotheses that match the Treebank-3 data, we also present the results using all available correctly tokenized ASR hypotheses. Our Joint-POS model consistently outperforms the pipeline *tri4* approach between 1.11% (*dev*, UAS) and 0.24% (*eval*, UAS) on the common subsets. The results are similar for the non-matching subsets. Note, that the results in Table 5 are for the small subset of utterances with a correct token sequence, i.e. where the (converted and filtered) Treebank-3 sentence tokens match the ASR hypothesis words exactly. This restriction allows an evaluation with LAS and UAS because the tokenization is identical and we have gold data for this correct token sequence. To (a) have a more extensive evaluation on all the utterances we have hypotheses for[13] and (b) be able to compare the pipeline and joint approach on the hypotheses coverage and close misses of the correct tokenization, too, we present Table 6.

We cannot use the standard parsing evaluation measures that depend on a correct word sequence to get scores on imperfectly recognized utterances.

[13]There are a few empty utterances with negligible counts.

We address this problem with a simple but imprecise solution: (1.) Parse the development and evaluation set using the parser models previously trained and tuned on the common sets (see Table 5); (2.) Evaluate the parser predictions on the ASR hypotheses against the gold Treebank-3 data with a imprecise scoring method that allows for a mismatch of the gold and predicted token sequence. We introduce two simple scores, unlabeled score (US) and labeled score (LS), with their names derived from UAS and LAS respectively (see Table 6). Recall that UAS requires a relation's head and dependent to match including their position and LAS requires a matching label (or dependency type) on that relation in addition.

The imprecision in the US and LS scoring stems from ignoring the positions of head and dependent in the utterance completely. We iterate over the utterances and for every token (or dependent) look up its head (word) and count this relation as a US match if the lookup is successful. When there is a US match, we also check for a matching label and count that as an LS match. The US and LS counts are normalized by the number of tokens in the Treebank-3 reference. The improvement our Joint-POS model shows over the pipeline *tri4* model is small for all scores, but consistent.

Model	Set	UAS	LAS	US	LS
tri4	dev	32.20	31.20	52.02	49.40
	eval	31.21	30.29	50.72	48.33
Joint-POS	dev	**32.41**	**31.43**	**52.21**	**49.71**
	eval	**31.56**	**30.73**	**51.21**	**48.99**

Table 6: Parsing results on full *dev* and *eval* sets. LAS, UAS, LS and US are given as percentages. The *dev* set has 3994 utterances with 44760 tokens and the *eval* set has 3912 utterances with 43277 tokens. Best scores per set in boldface.

5 DP-based analysis

We tentatively analyze in which cases the joint model does better than the pipeline approach. We first give absolute counts for how often this is the case in Table 7. While the Joint-POS model receives higher counts for all scores, there are also a considerable number of cases where the pipeline model makes fewer mistakes. We pick all examples randomly from the instances counted in the *All* column of Table 7 and focus on short sentences for presentability.

Model	UAS	LAS	US	LS	All
tri4	320	330	483	496	233
Joint-POS	332	363	540	596	267

Table 7: Utterance-based parsing evaluation. The numbers are counts of utterances where the model in the first column is better than the other. Column *All* gives the counts for when it is better on all four measures.

In the following examples, we highlight the important differences in boldface. In Figure 4, we see a fully correct Joint-POS model. While the pipeline approach does also recognize the correct word sequence, a POS tagging error causes the parsing to be erroneous on two arcs. This error affects all four scores (UAS, LAS, US and LS), as the parsing model not only misclassifies the label, but also attaches the head of "there" incorrectly. We visualize the error's effect in a correct vs incorrect tree comparison.

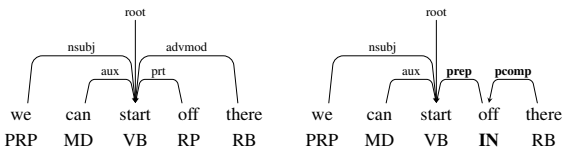

Figure 4: Dependency graph comparison #1. Correct Joint-POS tree on the left, incorrect *tri4* tree on the right.

We observe a recognition error in the pipeline *tri4* model that causes a different reading and syntactical structure in Figure 5. While it is acceptable spontaneous speech (e.g. "I like rock.. and like some country music."), "and" would not be the subject of the sentence.

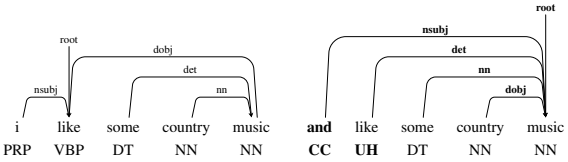

Figure 5: Dependency graph comparison #2. Correct Joint-POS tree on the left, incorrect *tri4* tree on the right.

The third graph visualization in Figure 6 illustrates an ASR deletion error on the first word. The pipeline *tri4* model handles the error gracefully, but receives lower US and LS scores because of the token mismatch nonetheless. If we had not allowed the imprecise evaluation, we would not have observed this kind of error.

The example in Figure 7 also has an ASR error in the pipeline approach at its core. In this case,

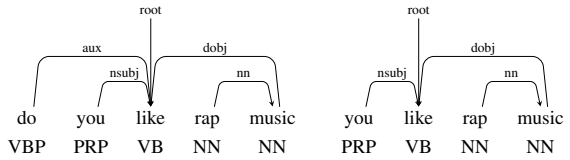

Figure 6: Dependency graph comparison #3. Correct Joint-POS tree on the left, incorrect *tri4* tree on the right.

while the joint model is entirely correct, the recognition error in the pipeline causes two POS tagging errors resulting in an incorrect parse.

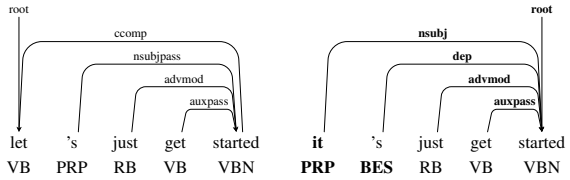

Figure 7: Dependency graph comparison #4. Correct Joint-POS tree on the left, incorrect *tri4* tree on the right.

The example utterance in Table 8 contains ASR errors in the both models' hypotheses with subsequent errors in POS tagging and parsing. We can glean that discourse interjections like "uh.. uh.." can be misrecognized as regular words, an error characteristic of spontaneous speech. Note, that the joint model gets the word "families" right, but as an object instead the subject. The pipeline model produces four word errors in sequence and "families" does not appear in its hypothesis.

6 Related work

Spoken language poses a variety of problems for NLP. The recognition of spoken language can suffer from poor recording equipment, noisy environments, unclear speech or speech pathologies. It also exhibits spontaneity, ungrammaticality and disfluencies, e.g. repairs and restarts (cf. Shriberg (1994)). Hence, in addition to ASR errors, downstream tasks such as parsing have to deal with these difficulties of conversational speech, whether the ASR output is in the form of n-best sequences or lattices. Jørgensen (2007) remove disfluencies prior to parsing and find their removal improves the performance of both a dependency and a head-driven lexicalized statistical parser on SWBD. In a more general joint approach of disfluency detection and DP, Honnibal and Johnson (2014) in contrast to Jørgensen (2007) make use of the disfluency annotations and report strong results for both, disfluency annotation and DP. Rasooli and Tetreault (2013) extend the arc-eager transition system (Nivre, 2008) with actions that handle reparanda, discourse markers and interjections, thereby also explicitly using marked disfluencies on SWBD for joint DP and disfluency detection. Where Rasooli and Tetreault (2013) and Honnibal and Johnson (2014) work with SWBD text data, Yoshikawa et al. (2016) are close to our setting and assume ASR output text as parser input. Yoshikawa et al. (2016) create an alignment that enables the transfer of gold treebank data to ASR output texts and add three actions to manage disfluencies and ASR errors to the arc-eager shift-reduce transition system of Zhang and Nivre (2011). While they do not parse lattices or confusion networks (lattices can be converted to confusion networks, see Mangu et al. (2000)) directly, Yoshikawa et al. (2016) use information from word confusion networks to discover erroneous regions in the ASR output. Charniak and Johnson (2001) parse SWBD after removing edited speech that they identify with a linear classifier. Additionally, Charniak and Johnson (2001) introduce a *relaxed edited* parsing metric that considers a simplified gold standard constituent parse (removed edited words are added back into the constituent parse for evaluation). Johnson and Charniak (2004) model speech repairs in a noisy channel model utilizing tree adjoining grammars (TAGs). Source sentence probabilities in the noisy channel are computed with a bigram LM and rescored with a syntactic parser for a more global view on the source sentence. The noisy channel is then formalized as TAG that maps source sentences to target sentences, where repairs are treated as the cleaned target side of the reparanda on the source side. Besides the words themselves, Johnson and Charniak (2004) use POS tags for the alignment of reparandum and repair, which indicates their usefulness in detecting disfluencies. Approaching spontaneous speech issues from another angle, Béchet et al. (2014) adapt a parser trained on written text by means of an interactive web interface (Bazillon et al., 2012) in which users can modify POS and dependency tags writing regular expressions.

Natural speech poses specific problems, but also comes with acoustic information that can improve parsing speech through its incorporation (Tran et al., 2017) or reranking (Kahn et al., 2005). Handling disfluencies following Charniak and Johnson

	Treebank-3				Joint-POS				*tri4*			
ID	Word	POS	Head	Dep.	Word	POS	Head	Dep.	Word	POS	Head	Dep.
1	well	UH	7	discourse	well	UH	7	discourse	well	UH	**0**	**root**
2	how	WRB	3	advmod	how	WRB	3	advmod	how	WRB	3	advmod
3	many	JJ	6	amod	many	JJ	**7**	**nsubj**	many	JJ	**1**	**dep**
4	uh	UH	6	discourse	**of**	**IN**	**3**	**dep**	**of**	**IN**	**3**	**dep**
5	uh	UH	6	discourse	**of**	**IN**	**3**	**prep**	**of**	**IN**	**3**	**prep**
6	families	NNS	7	nsubj	families	NNS	**5**	pobj	**own**	NNS	**5**	pobj
7	own	VBP	0	root	**own**	**VB**	0	root	**on**	**IN**	**3**	**prep**
8	a	DT	9	det	a	DT	9	det	a	DT	9	det
9	refrigerator	NN	7	dobj	refrigerator	NN	7	dobj	refrigerator	NN	7	**pobj**

Table 8: Example utterance. Errors in both models in boldface.

(2001), Kahn et al. (2005) rerank the n-best parses using a set of prosodic features in the reranking framework of Collins (2000). Kahn et al. (2005) find that combining prosodic features with non-local syntactic features increase F-scores in the relaxed edited metric of Charniak and Johnson (2001). Kahn and Ostendorf (2012) present an approach that automatically recognizes speech, segments a stream of words (e.g. a conversation side/speaker turn) into sentences and parses these. A reranker that can take into account ASR posteriors for n-best ASR hypotheses as well as parse-specific features for m-best parses can then jointly optimize towards WER (n hypotheses) or SParseval (Roark et al., 2006) ($n \times m$ hypotheses) metrics (Kahn and Ostendorf, 2012). Ehrlich and Hanrieder (1996) describe an agenda-driven chart parser that considers an acoustic word-level score from a word lattice and can combine a sentence-spanning analysis from partial hypotheses if a full parse is unobtainable. Tran et al. (2017) use speech and text domain cues for constituent parsing in an attention-based encoder-decoder approach based on Vinyals et al. (2015). They show that word-level acoustic-prosodic features learned with convolutional neural networks improve performance.

7 Discussion

Replacing words with word-POS pairs throughout the ASR process, as described in Section 3.1, increases the search space considerably. We focus on establishing the feasibility of this approach here and do not detail techniques to address this complexity issue. Including prior distributions of word-POS pair occurrences could help disambiguation early on in lattice creation. The LM in the joint model relies on word-POS pairs as well, and a smoothing approach that backs off to n-grams of words instead of n-grams of word-POS pairs would counter the increased sparsity due to the combination of words and their POS tags in the LM part. We only explore instances of errors the joint and pipeline models make in our analysis. A systematic error analysis identifying advantages and disadvantages of the joint model would be interesting, especially with the errors involving contractions and disfluencies. As a negative example for our joint model, we observed the separation of "didn't" as "did" plus "n't" as an ASR error for "did it". A qualitative analysis of error types could indicate whether this a random or systematic error, and the same is true of the positive examples in Section 5.

8 Conclusion

We have demonstrated a method to jointly perform POS tagging and ASR on speech. The tagging and parsing evaluations of the pipeline model vs our joint model confirm the successful integration of POS tags into speech lattices. While the improvements over the pipeline approach are small, we enrich lattices with POS tags that allow for latticed-based NLP in future work.

Acknowledgments

We thank the anonymous reviewers for their extensive and helpful feedback on this work. We also thank Xiang Yu for his parser implementation and Wolfgang Seeker for helping with the conversion to dependency parses. This work was funded by the German Research Foundation (DFG) through the Collaborative Research Center (SFB) 732, project A8, at the University of Stuttgart.

References

Tasos Anastasakos, John Mcdonough, Richard Schwartz, and John Makhoul. 1996. A compact model for speaker-adaptive training. In *Proc. ICSLP*, pages 1137–1140.

Thierry Bazillon, Melanie Deplano, Frederic Bechet, Alexis Nasr, and Benoit Favre. 2012. Syntactic annotation of spontaneous speech: application to call-center conversation data. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey. European Language Resources Association (ELRA).

Frédéric Béchet, Alexis Nasr, and Benoît Favre. 2014. Adapting dependency parsing to spontaneous speech for open domain spoken language understanding. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 135–139. ISCA.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly.

Sasha Calhoun, Jean Carletta, Jason M. Brenier, Neil Mayo, Dan Jurafsky, Mark Steedman, and David Beaver. 2010. The NXT-format switchboard corpus: a rich resource for investigating the syntax, semantics, pragmatics and prosody of dialogue. *Language Resources and Evaluation*, 44(4):387–419.

Sasha Calhoun, Jean Carletta, Daniel Jurafsky, Malvina Nissim, Mari Ostendorf, and Annie Zaenen. 2009. NXT switchboard annotations LDC2009T26. Web Download.

J.-C. Chappelier, M. Rajman, R. Aragüés, and A. Rozenknop. 1999. Lattice parsing for speech recognition. In *Proc. of 6ème conférence sur le Traitement Automatique du Langage Naturel (TALN99)*, pages 95–104, Cargèse (France).

Eugene Charniak and Mark Johnson. 2001. Edit detection and parsing for transcribed speech. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pages 118–126. Association for Computational Linguistics.

Stanley F. Chen and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393.

Yen-Lu Chow and Salim Roukos. 1989. Speech understanding using a unification grammar. In *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, pages 727–730 vol.2.

Christopher Cieri, David Graff, Owen Kimball, Dave Miller, and Kevin Walker. 2004. Fisher english training speech part 1 transcripts LDC2004T19. Web Download.

Christopher Cieri, David Graff, Owen Kimball, Dave Miller, and Kevin Walker. 2005. Fisher english training speech part 2, transcripts LDC2005T19. Web Download.

Christopher Collins, Bob Carpenter, and Gerald Penn. 2004. Head-driven parsing for word lattices. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain.*, pages 231–238. ACL.

Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, pages 175–182. Morgan Kaufmann.

Hal Daumé III. 2004. Notes on CG and LM-BFGS optimization of logistic regression. Paper available at `http://pub.hal3.name#daume04cg-bfgs`, implementation available at `http://hal3.name/megam/`.

Ute Ehrlich and Gerhard Hanrieder. 1996. Robust speech parsing. In *Proceedings of the Eight European Summer School In Logic, Language and Information*.

M.J.F. Gales. 1998. Maximum likelihood linear transformations for hmm-based speech recognition. *Computer Speech & Language*, 12(2):75–98.

John Godfrey and Edward Holliman. 1993. Switchboard-1 release 2 LDC97S62. Web Download.

John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Proceedings of the 1992 IEEE International Conference on Acoustics, Speech and Signal Processing - Volume 1*, ICASSP'92, pages 517–520, Washington, DC, USA. IEEE Computer Society.

Matthew Honnibal and Mark Johnson. 2014. Joint incremental disfluency detection and dependency parsing. *TACL*, 2:131–142.

Mark Johnson and Eugene Charniak. 2004. A tag-based noisy-channel model of speech repairs. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain.*, pages 33–39. ACL.

Fredrik Jørgensen. 2007. The effects of disfluency detection in parsing spoken language. In *Proceedings of the 16th Nordic Conference of Computational Linguistics NODALIDA-2007*, pages 240–244.

Jeremy G. Kahn, Matthew Lease, Eugene Charniak, Mark Johnson, and Mari Ostendorf. 2005. Effective use of prosody in parsing conversational speech.

In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 233–240, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jeremy G. Kahn and Mari Ostendorf. 2012. Joint reranking of parsing and word recognition with automatic segmentation. *Computer Speech & Language*, 26(1):1–19.

Faisal Ladhak, Ankur Gandhe, Markus Dreyer, Lambert Mathias, Ariya Rastrow, and Björn Hoffmeister. 2016. Latticernn: Recurrent neural networks over lattices. In *Interspeech 2016*, pages 695–699.

Linguistic Data Consortium. 2002. 2000 hub5 english evaluation speech LDC2002S09. Web Download.

Lidia Mangu, Eric Brill, and Andreas Stolcke. 2000. Finding consensus in speech recognition: word error minimization and other applications of confusion networks. *Computer Speech & Language*, 14(4):373–400.

Mitchell Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor. 1999. Treebank-3 LDC99T42. Web Download.

Robert Moore, Fernando Pereira, and Hy Murveit. 1989. Integrating speech and natural-language processing. In *Proceedings of the Workshop on Speech and Natural Language*, HLT '89, pages 243–247, Stroudsburg, PA, USA. Association for Computational Linguistics.

Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Comput. Linguist.*, 34(4):513–553.

B. T. Polyak and A. B. Juditsky. 1992. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. 2011. The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society. IEEE Catalog No.: CFP11SRW-USB.

Daniel Povey, Hong-Kwang Jeff Kuo, and Hagen Soltau. 2008. Fast speaker adaptive training for speech recognition. In *INTERSPEECH 2008, 9th Annual Conference of the International Speech Communication Association, Brisbane, Australia, September 22-26, 2008*, pages 1245–1248. ISCA.

Mohammad Sadegh Rasooli and Joel Tetreault. 2013. Joint parsing and disfluency detection in linear time. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 124–129, Seattle, Washington, USA. Association for Computational Linguistics.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, volume 1, pages 133–142. Philadelphia, USA.

B. Roark, M. Harper, E. Charniak, B. Dorr, M. Johnson, J. Kahn, Y. Liu, M. Ostendorf, J. Hale, A. Krasnyanskaya, M. Lease, I. Shafran, M. Snover, R. Stewart, and L. Yung. 2006. Sparseval: Evaluation metrics for parsing speech. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*. European Language Resources Association (ELRA).

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

David Ruppert. 1988. Efficient estimations from a slowly convergent robbins-monro process. Technical Report 781, Cornell University Operations Research and Industrial Engineering.

Elizabeth Ellen Shriberg. 1994. *Preliminaries to a Theory of Speech Disfluencies*. Ph.D. thesis, University of California at Berkeley.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *7th International Conference on Spoken Language Processing, ICSLP2002 - INTERSPEECH 2002, Denver, Colorado, USA, September 16-20, 2002*. ISCA.

Jinsong Su, Zhixing Tan, Deyi Xiong, Rongrong Ji, Xiaodong Shi, and Yang Liu. 2017. Lattice-based recurrent neural network encoders for neural machine translation. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 3302–3308. AAAI Press.

Keh-Yih Su, Tung-Hui Chiang, and Yi-Chung Lin. 1992. A unified framework to incorporate speech and language information in spoken language processing. In *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, volume 1, pages 185–188 vol.1.

Ann Taylor, Mitchell Marcus, and Beatrice Santorini. 2003. The penn treebank: An overview. In Anne Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, pages 5–22. Springer Netherlands, Dordrecht.

Trang Tran, Shubham Toshniwal, Mohit Bansal, Kevin Gimpel, Karen Livescu, and Mari Ostendorf. 2017. Joint modeling of text and acoustic-prosodic cues for neural parsing. *CoRR*, abs/1704.07287.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey E. Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2773–2781.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 323–333. The Association for Computer Linguistics.

Masashi Yoshikawa, Hiroyuki Shindo, and Yuji Matsumoto. 2016. Joint transition-based dependency parsing and disfluency detection for automatic speech recognition texts. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1036–1041. The Association for Computational Linguistics.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA - Short Papers*, pages 188–193. The Association for Computer Linguistics.