Sluice resolution without hand-crafted features over brittle syntax trees

Ola Rønning
University of Copenhagen
ronning@protonmail.com

Daniel Hardt
Copenhagen Business School
dh.digi@cbs.dk

Anders Søgaard
University of Copenhagen
soegaard@di.ku.dk

Abstract

Sluice resolution in English is the problem of finding antecedents of *wh*-fronted ellipses. Previous work has relied on hand-crafted features over syntax trees that scale poorly to other languages and domains; in particular, to dialogue, which is one of the most interesting applications of sluice resolution. Syntactic information is arguably important for sluice resolution, but we show that multi-task learning with partial parsing as auxiliary tasks effectively closes the gap and buys us an additional 9% error reduction over previous work. Since we are not directly relying on features from partial parsers, our system is more robust to domain shifts, giving a 26% error reduction on embedded sluices in dialogue.

1 Introduction

Sluices, also known as *wh*-fronted ellipses, are questions where the specification of what is asked for (beyond the *wh*-word), is elided (and thus needs to be retrieved from context). Below we distinguish two types of sluices: (i) embedded sluices, and (ii) root sluices. Embedded sluices occur in both single-authored texts and dialogue, while root sluices are particularly frequent in dialogue.

(1) If [this is not practical], explain *why*.

(2) A: [Jennifer is looking for you/~~me~~].
B: Why?

Example 1 is an embedded sluice. In it, *why* is the remnant of the embedded question, which we understand to mean '*why* this is not practical'. Example 2 is a root sluice. Again, *why* is the remnant of the question; however, the *wh*-word is not embedded in a larger structure. In both cases, we consider the antecedent of a *wh*-fronted ellipsis to be the content in the prior discourse that most intuitively provides the elided material, i.e., *[this is not practical]* in Example 1, and *[Jennifer is looking for ~~you~~/me]* in Example 2.[1]

Contributions This paper presents a more robust, neural model for sluice resolution in English based on multi-task learning. Our model significantly outperforms the only previous work on sluice resolution on available newswire corpora, but also has a number of advantages over this work. In particular, our model (a) does not require full syntactic parsing as a pre-processing step, (b) does not require manual feature engineering, and (c) is more robust when evaluated on speech corpora, because it is not dependent on full syntactic parsers (a). The lack of dependence on full syntactic parsers should also make it easier to transfer our model to new languages. In addition to the implementation of our architecture, which we make publicly available, we also make a new benchmark available for sluice resolution in English dialogue.

2 Related Work

Anand and McCloskey (2015) introduced the problem of sluice resolution and presented the newswire corpus which we use in our experiments below.

Anand and Hardt (2016) presented the first, and to the best of our knowledge only previous, sluice resolution system. They learn a linear combination of fifteen features across five feature groups, through a simple hill climbing procedure. Each

[1] In this work, we set aside cases where the discourse context does not provide an explicit antecedent.

Proceedings of NAACL-HLT 2018, pages 236–241
New Orleans, Louisiana, June 1 - 6, 2018.

feature is a score that represents a linguistic property defined over syntax trees. One feature group is *distance*, for example, which consists of various features encoding tree distances between candidate antecedents and the sluice. Candidates are restricted to be subtrees decorated with sentence labels. Note that this means that the model will ignore many candidates in domains where the syntactic parser is unable to identify full sentence subtrees. The other feature groups include: ii) *containment* of the sluice inside the candidate, iii) *discourse* structure encoding the discourse role of the candidate, iv) *content*, i.e., the semantic overlap between the candidate and the sluice, and v) *correlate*, i.e., semantic properties of the candidate, which may be predictive of sluice type (temporal, reason, degree, etc.). The linear model ranks all candidates and resolves a sluice by choosing the highest ranking candidate. Anand and Hardt (2016) use a slightly different metric than we do, because they rank syntactic subtrees that are potential antecedents, rather than labeling individual words in sequence. See §4. This paper is, to the best of our knowledge, the first to consider sluice resolution in dialogue, but Baird et al. (2018) consider sluice type classification in dialogue data.

Our work builds on recent progress in multi-task training of neural networks. Multi-task training of neural networks goes back to Caruana (1993), but was popularized by Collobert et al. (2011) and Søgaard and Goldberg (2016). The most common approach to multi-task training is to share all hidden parameters between different networks trained in parallel on different, but related datasets. The only requirement to the datasets is that they are defined in the same input space, and that there is a shared optimal hypothesis class for the shared parameters (Baxter, 2000), i.e., that there is a representation that is optimal for all the related tasks in question. Obvious extensions to this approach include sharing only parameters in specific layers (Søgaard and Goldberg, 2016; Misra et al., 2016), subspaces (Bousmalis et al., 2016), or doing only soft sharing (Duong et al., 2015), i.e., penalizing the ℓ_p distance between the models.

In addition to a single-task recurrent neural network baseline, we use the approach in Søgaard and Goldberg (2016) where only initial layers are shared, as our baseline. Our approach to sluice resolution is largely inspired by the network architecture in Hashimoto et al. (2016).

3 Our approach

Our approach is an extension of previous work on multi-task learning, largely inspired by Hashimoto et al. (2016). We construct a neural architecture based on recurrent neural networks (Hochreiter and Schmidhuber, 1997), which differ only from the architectures discussed above in using label embeddings that are also passed on to subsequent layers, skip connections from the embedding layer, and regularization. The stacking on label embeddings from auxiliary tasks makes our approach similar to stacked learning (Wolpert, 1992) and progressive neural networks (Rusu et al., 2016).

Unlike Hashimoto et al. (2016), we do not optimize for a joint optimum, only for sluice resolution performance. The architecture that performs best on development data has two interesting properties: (a) It was also the architecture that converged the fastest. (b) It induces a linguistically motivated ordering of the auxiliary tasks in terms of abstractness. The architecture learns part of speech (POS) tagging at the initial layer; then syntactic chunking, then combinatory categorial grammar (CCG) supertags, before learning sluice resolution at the outer layer. See Figure 1 for a diagram of our architecture. We train our architecture by sampling from all our tasks with equal probability. The instance loss is computed at the appropriate level of the network, and backpropagation will only affect the previous levels. All our neural networks use 50 dimensional pre-trained GloVe embeddings, trained by (Pennington et al., 2014) on Wikipedia and Gigaword 5. The word embeddings are *not* updated during training. Similarly, all our networks were trained for 30 epochs. They all use ZoneOut (Krueger et al., 2016) regularization with Z-state 0 and Z-cell 0.2 (except the single-task baseline, which used Z-cell 0.0), batches of 10 examples and are optimized using the Adam optimizer (Kingma and Ba, 2014) with initial learning rate 0.001 (except the single-task baseline, which used a learning rate of 0.01). All LSTMs contain 64 hidden units. All additional hyper-parameters were tuned manually.

4 Experiments

Corpora We evaluate our models on two datasets, the newswire corpus introduced in Anand

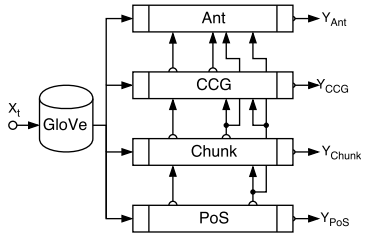

Figure 1: Our architecture. *Ant* is for sluice antecedent tagging.

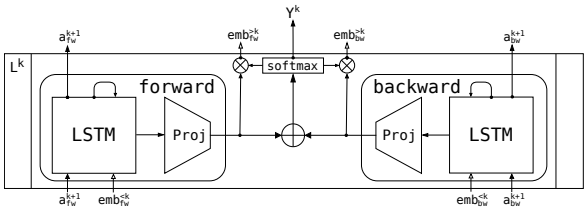

Figure 2: Graphical representation of layer L^k in our architecture. L^k solves task k. Input to L^k: Word w_t embedded into $\mathbb{R}^n$, activation a_t^{k-1}, and label embeddings $emb_t^{<k}$. L^k outputs: estimated label $\hat{y}_t^k$, embedding of $\hat{y}_t^k$ in $\mathbb{R}^H$ and LSTM activations a_t^k.

and McCloskey (2015) (ESC) and a novel corpus of annotated sluices, which is a small subset of the English part of the OpenSubtitles corpus (Tiedemann, 2009). All models are trained on ESC and evaluated on both datasets.

ESC consist of 3103 annotated examples of embedded sluices in written language. The sluices were collected in the New York Times section of English Gigaword. The annotations provide us with the antecedent, a paraphrasing without *wh*-ellipsis, and automatically obtained syntactic trees. We follow Anand and Hardt (2016) in treating the first annotator in each example as the gold-standard.

To measure the sensitivity of our systems to domain shifts, we annotate a total of 2000 examples from the OpenSubtitles corpus. 1000 examples are root sluices, and 1000 are embedded sluices. Each example is annotated by two annotators. Inter-annotator scores were 0.77 for embedded sluices, and 0.83 for root sluices.

Auxiliary Tasks We use four auxiliary tasks in our experiments below:

POS tagging is the task of determining the syntactic category (part of speech) of a word in context. Our data is from the Wall Street Journal section of the English Penn Treebank, using the splits in the CONLL 2007 shared task (Nivre et al., 2007).

Chunk-ing is a partial parsing task in which we need to identify the boundary of the main phrases in a sentence. Our data is from the 2000 CoNLL shared task (Tjong Kim Sang and Buchholz, 2000).

Com Sentence compression is the task of sentence parts that can be dropped without loosing coherence nor salient information. We use the dataset also used in (Knight and Marcu, 2000).

CCG super-tagging is another form of partial parsing, using a more fine-grained tagset. We use the CCGBank with standard splits.[2]

The Søgaard and Goldberg (2016) model uses sentence compression at the lowest layer, then chunking, and finally antecedent tagging at the highest.

We observed a detrimental effect when including compression in the same stack as the other auxiliaries for the model presented here. This effect vanished when compression is placed in a separate stack.

Evaluation metrics We evaluate predicted antecedents using (token-level) F1 scores. This metric is motivated by the observation that annotated spans vary in length, and that annotators often disagree about the exact bracketing; it differs from the one used in Anand and Hardt (2016), however, and we stress that our results are therefore not directly comparable to those reported in their paper. Moreover, Anand and Hardt (2016) used cross-validation; we compare systems and baselines on a fixed split.

Baselines In addition to comparing to Anand and Hardt (2016), the only previous work on sluice resolution, we compare our performance to two baseline neural network architectures: a single-task architecture and a multi-task architecture similar to Søgaard and Goldberg (2016).

Our first baseline is a single-task, two-layered long-short-term memory (LSTM) network, with

[2] http://groups.inf.ed.ac.uk/ccg/ccgbank.html

	NEWSWIRE	DIALOGUE	
Model	Embedded	Embedded	Root
Anand and Hardt (2016)	0.67	0.23	0.06
Single-task baseline	0.54	0.41	**0.28**
Søgaard and Goldberg (2016)	0.64	0.41	0.20
This work	**0.70**	**0.51**	0.17

Table 1: F1 scores on embedded sluices from ESC (Newswire) and embedded and root sluices from OpenSubtitles (Dialogue).

a projection layer and a softmax layer. Our second baseline is a cascading, three-layered LSTM, as described by (Klerke et al., 2016). See §3 for hyper-parameters.

Replicability We make our corpus splits, our annotations, our final models, and our source code available at `https://github.com/OlaRonning/sluice_antecedent_selection`.

5 Results

Scores are listed in Table 1. We first observe that using multi-task learning closes the gap between our neural network baselines and previous work, providing a new state-of-the-art for sluice resolution. We also note that our model converges on the validation set after only 5 epochs, as compared to 20-25 epochs for our neural baseline architectures.

Moving from newswire to dialogue, the gap between our system and previous work widens. This indicates that our architecture is much more robust to domain shifts than previous work. Our neural baselines also do better than previous work when doing evaluation in a cross-domain setup.

All systems perform significantly worse on out-of-domain data than on newswire. In particular, we see all models struggle with root sluices. Here, interestingly, our single-task baseline actually performs best of all systems, with a token-level F1 score of 0.28.

6 Error analysis

Previous work is sensitive to parse quality Our most important observation in our error analysis is that the system by Anand and Hardt (2016) is very sensitive to the quality of the syntactic parse trees. If we consider only test examples where the antecedent forms a syntactic constituent, according to the error prone parse tree, Anand and Hardt (2016) achieve a token-level F1 score of 0.81. Antecedents need not, but are generally expected to be syntactic constituents, so the lower performance on the rest of the examples (token-level F1 0.53) is likely due to errors introduced by the syntactic parser.

Long distance sluice resolution is hard Both previous work and all our neural systems perform relatively well on examples where the distance between sluice and antecedent is short, e.g., one or two sentences, but none of the systems are good at resolving sluices with three or more sentences between sluice and antecedent. These cases are very rare, about one percent, in ESC, and we leave long distance sluice resolution as an open research problem for now.

Dialogue is harder - root sluices, in particular We also note that some errors in the dialogue corpus derive from examples where the sluices do not have *any* antecedents in the dialog. Here, instead, physical interactions trigger *wh*-fronted ellipses; see Example 3, for example:

(3) *A enters room*
B: What ~~do you want~~ ?

In order to resolve such examples, we would need to use multi-modal input and learn from both visual and auditory cues.

7 Conclusion

We have presented a neural architecture for English sluice resolution and shown that it outperforms previous work on sluice resolution. Our approach also has several advantages over previous work; most importantly, not relying on handcrafted features over full syntactic trees. Instead

we use multi-task learning to induce syntactic information in a way that does not require access to syntactic information at test time. Not conditioning on features defined over brittle syntax trees also makes our approach less vulnerable to domain shifts. In order to show this, we annotate a new benchmark dataset for sluice resolution in English spoken language. On spoken language data, the gap between our architecture and previous work widens significantly. That said, sluice resolution in spoken language is much harder than sluice resolution in newswire for models trained on newswire; and all the models in our experiments found it particularly hard to resolve root sluices as opposed to embedded ones. Our error analysis also indicates that long distance sluice resolution remains an open problem.

Acknowledgments

We would like to thank Anissa Hamza for her assistance with annotating, and Austin Baird for help with the AntRank system. Thanks also to Pranav Anand and Jim McCloskey for help with the sluicing newswire data. Anders Søgaard was supported by the European Research Council and the Innovation Fund Denmark. Daniel Hardt was supported by the U.S. National Science Foundation, grant 1451819.

References

Pranav Anand and Daniel Hardt. 2016. Antecedent selection for sluicing: Structure and content. In *EMNLP*. pages 1234–1243.

Pranav Anand and Jim McCloskey. 2015. Annotating the implicit content of sluices. In *Proceedings of The 9th Linguistic Annotation Workshop*. Association for Computational Linguistics, Denver, Colorado, USA, pages 178–187. http://www.aclweb.org/anthology/W15-1621.

Austin Baird, Anissa Hamza, and Daniel Hardt. 2018. Classifying Sluice Occurrences in Dialogue. In *LREC*.

Jonathan Baxter. 2000. A model of inductive bias learning. *Journal of Artificial Intelligence Research* 12:149–198.

Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. 2016. Domain Separation Networks. In *Proceedings of NIPS*.

Rich Caruana. 1993. Multitask learning: a knowledge-based source of inductive bias. In *ICML*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research* 12:2493–2537.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Low Resource Dependency Parsing: Cross-lingual Parameter Sharing in a Neural Network Parser. In *Proceedings of ACL*.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2016. A joint many-task model: Growing a neural network for multiple nlp tasks. *arXiv preprint arXiv:1611.01587* .

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Sigrid Klerke, Yoav Goldberg, and Anders Søgaard. 2016. Improving sentence compression by learning to predict gaze. *arXiv preprint arXiv:1604.03357* .

Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization-step one: Sentence compression. *AAAI/IAAI* 2000:703–710.

David Krueger, Tegan Maharaj, János Kramár, Mohammad Pezeshki, Nicolas Ballas, Nan Rosemary Ke, Anirudh Goyal, Yoshua Bengio, Hugo Larochelle, Aaron Courville, et al. 2016. Zoneout: Regularizing rnns by randomly preserving hidden activations. *arXiv preprint arXiv:1606.01305* .

Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. 2016. Cross-Stitch Networks for Multi-Task Learning. In *Proceedings of CVPR*.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The conll 2007 shared task on dependency parsing. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. pages 1532–1543.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671* .

Anders Søgaard and Yoav Goldberg. 2016. Deep multitask learning with low level tasks superviser at lower layers. In *ACL*.

Jörg Tiedemann. 2009. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria, volume V, pages 237–248.

Erik F Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the conll-2000 shared task: Chunking. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7*. Association for Computational Linguistics, pages 127–132.

David Wolpert. 1992. Stacked generalization. *Neural Networks* 5:241–259.