Limited memory incremental coreference resolution

Kellie Webster and **James R. Curran**
ə-lab, School of Information Technologies
University of Sydney
NSW 2006, Australia
{kellie.webster,james.r.curran}@sydney.edu.au

Abstract

We propose an algorithm for coreference resolution based on analogy with shift-reduce parsing. By reconceptualising the task in this way, we unite ranking- and cluster-based approaches to coreference resolution, which have until now been largely orthogonal. Additionally, our framework naturally lends itself to rich discourse modelling, which we use to define a series of psycholinguistically motivated features. We achieve CoNLL scores of 63.33 and 62.91 on the CoNLL-2012 DEV and TEST splits of the OntoNotes 5 corpus, beating the publicly available state of the art systems. These results are also competitive with the best reported research systems despite our system having low memory requirements and a simpler model.

1 Introduction

Coreference resolution is the task of partitioning mentions in a document, usually noun phrases, into clusters which correspond to their real world referents. It is typically approached as a classification task between mentions; given a set of mentions, systems predict the likelihood of their being coreferential with one another and translate these scores into a clustering in a decoding phase.

The task has received considerable research attention due to its importance for downstream inference in tasks such as named entity linking and relation extraction. While simple, local models of coreference have established a reasonable baseline, encoding global consistency requirements remains a challenge since their complete representation is computationally intractable. Two promising but orthogonal directions addressing the need for global consistency measures are ranking-based decoding (Ng and Cardie, 2002; Denis et al., 2007; Fernandes et al., 2012; Durrett and Klein, 2013; Chang et al., 2013) and cluster-based modelling (Rahman and Ng, 2009; Raghunathan et al., 2010; Lee et al., 2011; Klenner and Tuggener, 2011). However, among current systems, decoding strategies are increasingly complex and cluster-based models do not fully leverage psycholinguistic cues such as reading order.

The primary contribution of our work is a reconceptualisation of the coreference task by analogy with the shift-reduce parsing algorithm. This reconceptualisation allows us to capitilise on both ranking- and cluster-based approaches and our system, LIMERIC, outperforms systems using either approach in isolation. We go beyond the shift-reduce algorithm by interpreting our stack of partially formed clusters as a reader's mental status while reading. This allows us to introduce a series of rich discourse features which capture antecedent competition and cognitive accessiblity via a cluster's position in the stack.

Our system is simple and efficient, using maximum-margin averaged perceptron classification and optional beam-search decoding during inference. Despite requiring only a limited amount of memory, our system achieves the competitive CoNLL scores of 63.33 and 62.91 on the CoNLL-2012 DEV and TEST splits of the OntoNotes 5 corpus (Pradhan et al., 2012). We argue that this is due to its more faithful representation of cognitive processing and that extending psycholinguistic insights in modelling is a very promising research direction for even further improvement.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 2129–2139, Dublin, Ireland, August 23-29 2014.

2 Related work

Early computational approaches to coreference resolution were built around what is now referred to as mention-pair models. Such models use two stage resolution; the first stage calculates pairwise scores reflecting the likelihood that a mention and its candidate antecedents are coreferential while the second phase decodes these scores into coreference clusters. The simplest way to decode is locationally greedy (Soon et al., 2001), in that the closest candidate with a compatibility score over some threshold is deemed a mention's antecedent. Anaphoricity determination (determining whether a mention constitutes a good first mention of an entity) is mediated by the threshold since a mention without a sufficiently good candidate antecedent starts a new cluster. While these local models achieve a reasonable baseline (Soon et al. (2001) achieves MUC F-scores of 62.6 and 60.4 on MUC 6 and 7), they can make global consistency errors which limit their usefulness downstream. For instance, in the following excerpt from `bn/voa/00/voa_0068` of OntoNotes 5, it is possible that a system uses local evidence such as synonymy to misclassify the ship as the antecedent of a huge Norwegian transport vessel and similarly The battered US Navy destroyer Cole as the antecendent of the ship; unfortunately, these local decisions imply a clustering in which Cole is referred to as a Norwegian transport vessel.

> **The battered US Navy destroyer Cole** has begun **its** journey home from Yemen ... Flanked by other US warships and guarded by aircraft, **the ship** was towed out of Aden Harbor to rendezvous with **a huge Norwegian transport vessel**

While exhaustive comparison would remedy the situation, complete inference has exponential time complexity and so is unrealistic for practical systems. Furthermore, since humans are able to resolve reference on the fly, it seems reasonable that psycholinguistic heuristics would similarly help the task while remaining efficient.

Active research aims to approximately encode global consistency measures, via ranking-based decoding and cluster-level modelling. Ranking-based decoding strategies (Ng and Cardie, 2002; Denis et al., 2007) improve locationally greedy decoding by defining a search window and deeming the best, rather than the closest, candidate within the window to be a mention's antecedent. The publicly available Reconcile system[1] (Stoyanov et al., 2010a; Stoyanov et al., 2010b) uses a simple encoding of this strategy while more recent approaches (Fernandes et al., 2012; Durrett and Klein, 2013; Chang et al., 2013) incorporate the concept within highly sophisticated models. While these systems achieve state of the art performance, they do so at the expense of model complexity.

In cluster-level modelling approaches (Rahman and Ng, 2009; Raghunathan et al., 2010; Lee et al., 2011; Klenner and Tuggener, 2011), instead of basing scoring on the compatibility of pairs of mentions, mentions are compared against incrementally grown partial clusters. This, for instance, may allow a huge Norwegian transport vessel to be compared against a cluster containing both the ship and The battered US Navy destroyer Cole, allowing nationality discord to weigh against the clustering. In this way, global consistency information becomes more important as a mention needs to be compatible with multiple mentions in a cluster, rather than its closest or best antecedent. However, there have been problems with these implementations including their being heavily focussed on surface level features and failing to fully utilise psycholinguistic cues such as reading order. A notable exception is Recasens et al. (2013), which provides a computational model of low salience discourse entities and demonstrates its efficacy in filtering system mentions in the Stanford sieve system (Raghunathan et al., 2010; Lee et al., 2011).

Consistent with Klenner and Tuggener (2011) and others, we argue that psycholinguistic insight is the key to unite cluster- and ranking-based models. This is because theories such as Centering Theory (Grosz et al., 1995) and Accessibility Theory (Ariel, 2001) describe how the human mind keeps track of discourse referents as entities rather than distinct mentions, and resolves anaphora via ranked cognitive accessibility. By reformulating the coreference resolution by analogy with the shift-reduce parsing algorithm, we gain access to the stack of active discourse entities which we rank in order of salience. In this way, the stack in our model becomes an approximation of a reader's mental state when reading a document, allowing us to directly model cognitive models of discourse.

[1] `http://www.cs.utah.edu/nlp/reconcile`

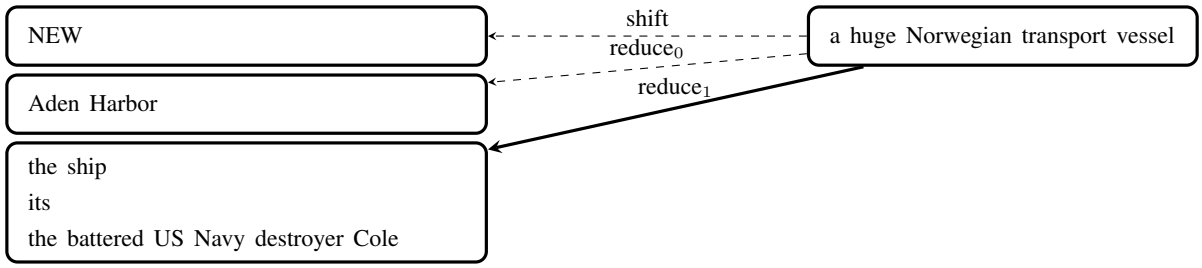

Figure 1: shift-reduce comparisons, `bn/voa/00/voa_0068`

3 Psycholinguistically informed coreference resolution

The shift-reduce algorithm (Aho and Johnson, 1974) is widely used in parsing due to its efficiency, the simplicity of its data structures, and its limited memory usage. For syntactic parsing, a queue is initialised with a series of tokens which is processed in a single reading order pass. Tokens either *shift* onto a stack as a leaf fragment or *reduce* with an existing fragment to former a larger phrasal unit. For the reduce operation, the classifier needs to determine into which fragment the token should merge.

By drawing an analogy between tokens and phrases in syntactic parsing with mentions and clusters in coreference resolution, we derive an algorithm for the latter. In particular, we can initialise a queue of mentions and maintain a stack of clusters which incrementally grow as we read a document. Our classifier is similarly tasked with determining whether a mention should shift onto the stack as the first mention of a new discourse entity or should reduce with an already active one (see Figure 1 for the example of resolving the enqueued mention a huge Norwegian transport vessel). For reduce operations, we additionally need to determine into which entity cluster the mentions should merge. In this way, the shift-reduce algorithm expresses a joint decision of anaphoricity and coreferentiality.

We draw from shift-reduce parsing its simplicity and small memory requirement since we believe these give rise to a more faithful representation of cognitive processing. There are, however, some technical points to consider. Instead of the reduce operation applying to a small window at the top of the stack (top two in the case of binarised grammars), we want to search potentially the whole stack, as described the general formulation of the algorithm. While a full search gives our process worst case $O(n^2)$ time complexity, this is only occurs in the case of an incoherent document which mentions each of its discourse entities exactly once. In the average case, exhaustive stack search still represents a time saving compared to full mention-pair models which compare each mention against all potential antecedent *mentions*. Also, we don't aim to form a single full tree covering all the mentions but rather a collection of clusters. While it is possible to define a document graph of coreference relations (as demonstrated in Fernandes et al. (2012)), it is not necessary to do so.

The algorithms we employ for training and inference our system are represented in Figure 2.

Initialisation

We initialise the stack to be empty and the queue to be the complete set of mentions extracted from the parse structure and named entities in a document. Following the literature, our mention extraction module is designed to be high recall since missed mentions are guaranteed to hurt performance, while it is possible to learn that spurious mentions should not be reported (e.g. Durrett and Klein (2013)). Thus, we train and test on predicted mentions despite the availability of gold mentions for training (to keep system input as similar as possible between training and testing environments) and at test time (since this is not realistic). In this way, we learn a model that is robust to noise in mention extraction.

Learning

On each training pass through a document, we read the enqueued mentions exactly once, in reading order without look ahead. As each mention comes to head the queue, we generate a training instance in which the classifier decides whether it is more likely that the mention *shift* onto the stack as the first mention of a new discourse entity or *reduce* with the cluster of an already active one. In particular, the reduce score

```
initialise queue;
initialise stack;
while queue do
    active = queue.pop();
    prediction = classify(active, stack);
    gold = correct_classification(active, stack);
    if prediction != gold then
        update(prediction, gold);
    end
    cluster = apply_pred(active, stack, gold);
    promote(cluster, stack);
end
```

```
initialise queue;
initialise stacks;
while queue do
    active = queue.pop();
    forall stacks do
        prediction = classify(active, stack);
        cluster = apply_pred(active, stack, prediction);
        promote(cluster, stack);
    end
    prune_stacks(stacks);
end
```

Figure 2: learning (left) and inference (right) algorithms

is the highest of all potential merges. Features are generated on the fly to reduce memory requirements, and because the state of the system is determined by each move made. The margin of classification is widened by augmenting by one the scores corresponding to non-gold decisions.

We then determine whether any difference exists between the classifier's decision and the gold answer key by looking for one of five errors, three taken from Durrett and Klein (2013) (falsely anaphoric, falsely new, wrong link made) and two inspired by the categories used in Kummerfeld and Klein (2013) (extra mention and extra entity). If an error is detected, we perform perceptron updates of the feature weights, increasing those corresponding to the gold decision and decreasing those corresponding to the incorrect prediction. We find that varying the feature value update according to the error made has a performance benefit, particularly when 'falsely new' is given a faster learning rate. This may due to sparsity: across a corpus, the number of first mentions of an entity is smaller than both that of subsequent mentions, and singleton mentions. We note that it should be possible to learn a model using uniform updates by increasing the number of training iterations, though this increases the chance of overfitting. Also, tuning these parameters may affect different balances in error types for different applications.

As noted in Rahman and Ng (2009), since the mention-cluster indicator functions do not apply to the case where a new entity is formed (shift operations), reduce comparisons activate many more features than shift ones do. During development, we noticed that this marked difference in feature set size was negatively impacting performance as reduce operations were unfairly favoured. To grow the shift feature weights faster, we introduced a scaling parameter on the update of these feature weights; we found the ratio of the feature space sizes to work well.

As the final stage, the system applies the decided move. There are two valid 'decided' moves, namely the correct decision, read from the gold standard, or the (potentially incorrect) predicted decision. In this work, we train by following the path of correct decisions, though we plan future research implementing the latter. We hope this will improve the robustness of our system given analogous findings in shift-reduce parsing (Zhang and Nivre, 2012). Novel to our approach, the cluster resulting from application of the decided move is promoted to the top of the stack since recency increases cognitive accessibility. This is a crucial implementation detail given the cognitive interpretation we give to the stack of clusters.

Inference

A benefit of our formulation of the coreference task is that inference is little different to training, without feature weight tuning. In both, documents are processed via a queue of mentions, though a single stack is replaced by possibly multiple in a beam regularly pruned to a fixed width. This has possible cognitive underpinnings since humans need to be able to back track if an interpretation proves incorrect. Analogously, it allows our system to reduce the impact of potentially harmful local decisions. Interestingly, we find in Section 6 that this has little appreciable impact on performance, though this is consistent with Zhang and Nivre (2012), which finds that beam search in inference can hurt the performance of a shift-reduce syntactic parser trained on gold decisions.

4 Rich discourse features

We base our feature space on the pool of features described in the literature (Soon et al., 2001; Ng and Cardie, 2002; Bengtson and Roth, 2008; Stoyanov et al., 2010a; Stoyanov et al., 2010b; Raghunathan et al., 2010; Lee et al., 2011). We introduce *Discourse likelihood* as a novel extension of work in Recasens et al. (2013), designed to mediate system conservativeness in the decision between whether a cluster remains a singleton or grows into a larger cluster. Conjunct discord is introduced as a finer grained extension of traditional number agreement features.

If existing features apply to single mentions or single clusters, they apply in this same way in our system. To map the mention-pair features in the literature into functions which take a mention-cluster pair, we use a range of strategies including the existence of a compatible mention for the active among the cluster pool (Raghunathan et al., 2010), binned proportion of clustered mentions compatible with the active mention (Rahman and Ng, 2009), or the maximum compatibility score between the clustered mentions and the active one (based on Ponzetto and Strube (2006)).
The examples here correspond to the reduce$_1$ move in Figure 1.

Lexical data driven lexicalised features from Durrett and Klein (2013); for the active mention the ship, we would generate the features like **`head_word:ship first_pos:DT last_shape:LOWER`**

String match existence and proportion of clustered mentions with various string matches with the active mention, e.g. **`head_match:none acronym:none`**

Attribute agreement agreement in animacy, gender, number, and NER values pooled across the cluster, and active, e.g.**`number_agree:true`**

Attribute discord where mentions are conjunctions, disagreement between the number of sibling NP children; disagreement between the citation form of any pronouns in cluster and active

Syntax existence of i-within-i (restriction on anaphora due to government and binding requirements on a sentence's parse tree) or subject-object relation between active and any clustered mention e.g.**`iwithini:none`**

Semantics binned value of maximum Lin et al. (2012) similarity score between active and clustered mention heads, e.g. **`lin:high`** since ship and vessel are highly related; disagreement between coarse grained semantic classes of nominals determined from WordNet (Fellbaum, 1998)

Length length of mention in number of tokens **`mlength:3`**; length of cluster in number of mentions

Distance distance between active and closest clustered mention, measured in number of sentences and number of intervening mentions

Discourse patterns whether any subsequent mention is an indefinite nominal

Discourse likelihood an integer value representing the likelihood that cluster has proposed length (singleton or not) given the internal morphosyntactics of the clustered mentions; likelihood of stack given likelihood of contained clusters

4.1 Stack features

Since position in stack in our model represents relative cognitive accessibility, we introduce Depth features as the cognitive analogues of Distance features, designed to more faithfully represent accessibility.

Stack depth depth from top of the stack, binned as top cluster in stack, within five clusters from the top, within ten clusters from the top, outside this[2]; raw depth, depth normalised in turn by ignoring singletons and ignoring clusters not containing a proper name mention **`raw_depth:high`**, **`ne_depth:top`** were all used, with the last two designed to capture the impact of salience

[2]these values were empirically optimised, though we note that they reflect known constraints on human short-term memory

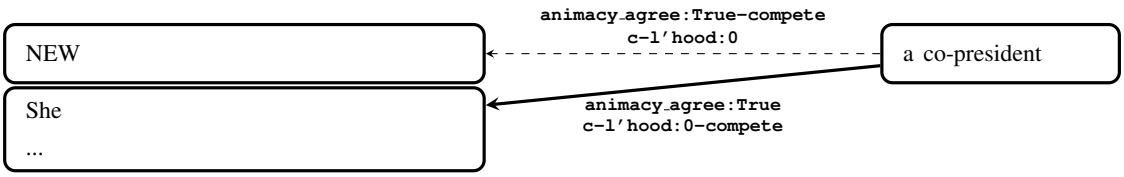

Figure 3: stack features, bn/mnb/00/mnb_0023

Stack competition In an aim to model the competition between clusters in the stack, we introduce stack competition features. In addition to evaluating our mention-cluster features between the active mention and its proposed antecedent, we also evaluate them between it and the other clusters comprising the stack. If a stacked cluster evaluates positively with any of these, we generate a labelled version of the indicated feature. By having these features compete with those of the proposed antecedent, we hope to better learn a more global ranking of candidates than straightforward search window strategies do. Figure 3 shows how stack features can be used to distinguish between the attractiveness of merging an indefinite nominal into a cluster (attractive due to matching linguistic attributes) as compared to starting a new discourse entity (attractive due to discourse likelihood of indefinite nominal in a new cluster).

4.2 Discourse transition prefixing

After Durrett and Klein (2013), we use discourse transition strings formed from the types of the mention and its closest candidate antecedent as feature prefixes, e.g. **m:nominal+a:nominal**. While this inflates the potential size of the feature space[3], the features generated are more meaningful since we would expect many indicator functions to behave differently for pronouns than for subsequent proper names, for example, reintroducing entities. Also, since we use perceptron learning, feature weights are only tuned if the feature is useful in making a decision during training.

5 Results

We evaluate LIMERIC on the OntoNotes 5 corpus (Pradhan et al., 2012) with the included parse and NER annotations. Our experimental setup matches the specifications of the CoNLL-2012 shared task: we use the standard corpus splits, official scorer, and report performance on the CoNLL metric which averages the MUC F-score (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998) and CEAFE (Luo, 2005).

We compare our performance against that of three state of the art systems which reflect the diversity of current approaches. Stanford[4] (Lee et al., 2011) has rule-based decoding with cluster-based modelling. Berkeley[5] (Durrett and Klein, 2013) uses mention-pair features in a factor graph model. Since reported performance for this system is on CoNLL-2011, we compare against the publicly available system using the SURFACE model, which doesn't use features induced from English Gigaword (Graff et al., 2007). Chang et al. (2013)'s L^3M systems comprise both mention-pair and cluster-based variants; we focus on the former here since these perform better on OntoNotes 5. L^3M represents a maximum-margin approach to ranking models, where CL^3M adds some cluster modelling via a constraint term.

5.1 Performance

Table 1 presents our performance on DEV and TEST. Our core LIMERIC system includes all features described in Section 4 including our novel discourse features Discourse patterns, Discourse likelihood, and Stack depth. In development, we experiment with system configurations by deactivating semantic features (-*s*) and activating stack competition features (+*c*) in turn. Despite a good CEAFE score, we opt not to include stack competition features in our final system.

Given the simplicity of our learning and decoding, our system compares favourably with existing systems. In all configurations, we beat both publicly available systems and the mention-pair variant L^3M: by uniting aspects of ranking- and cluster-based approaches, we achieve benefits beyond either in

[3] since distinct feature strings correspond to completely distinct features

[4] http://nlp.stanford.edu/software/corenlp.shtml

[5] http://nlp.cs.berkeley.edu/berkeleycoref.shtml

System	MUC	B^3	CEAFE	CoNLL
Stanford	64.30	70.46	46.35	60.37
Berkeley	66.10	68.56	50.09	61.58
L^3M	67.88	71.88	47.16	62.30
CL^3M	69.20	**72.89**	48.67	**63.59**
LIMERIC	**71.02**	68.66	50.31	63.33
LIMERIC +*c*	70.67	68.33	**50.55**	63.18
LIMERIC -*s*	70.53	68.21	50.34	63.03

System	MUC	B^3	CEAFE	CoNLL
Stanford	63.83	68.52	45.36	59.23
Berkeley	69.09	65.89	48.26	61.08
L^3M	68.31	70.81	46.73	61.95
CL^3M	69.64	**71.93**	48.32	**63.30**
LIMERIC	**71.52**	67.47	**49.75**	62.91

Table 1: CoNLL-2012 DEV (left) and TEST (right)

isolation. Also, we consistently outperform CL^3M on two of the three performance metrics; our method for uniting existing approaches is more direct and psycholinguistically faithful than that in CL^3M and our competitive system results are promising for future work.

Our system's MUC and CEAFE scores are the highest across all systems on both datasets. Our high CEAFE score in particular suggests that our system produces an accurate *number* of clusters. We explore this further in Figure 4 using the tool described in Kummerfeld and Klein (2013)[6]. Between Berkeley and LIMERIC, the notable difference is that we make considerably fewer Divided Entity and Missed Entity errors for a small increase in Conflated Entity errors. By introducing features which model when a new discourse entity should form and how the relative accessibility of already active ones impacts coreference decisions, we more accurately predict the bounds of entity clusters. This modelling is independent of surface features: 85% and 96% of Berkeley's Divided Entity errors occur where there is no head match and string match between mentions, respectively, compared to our values of 87% and 96%.

We note also that, given Kummerfeld and Klein's finding that MUC recall is highly sensitive to Divided Entity errors and B^3 precision to Conflated Entity errors, we can understand our performance on these metrics, particularly if our errors occur in larger clusters.

Between LIMERIC and LIMERIC+*c*, the notable difference is that LIMERIC+*c* makes fewer Missed Mention errors, but at a high cost to Extra Entity errors. A principled solution for future work might be to enrich our model of what makes a discourse transition unfavourable, in contrast to the predominate tradition of modelling what makes a discourse transition favourable.

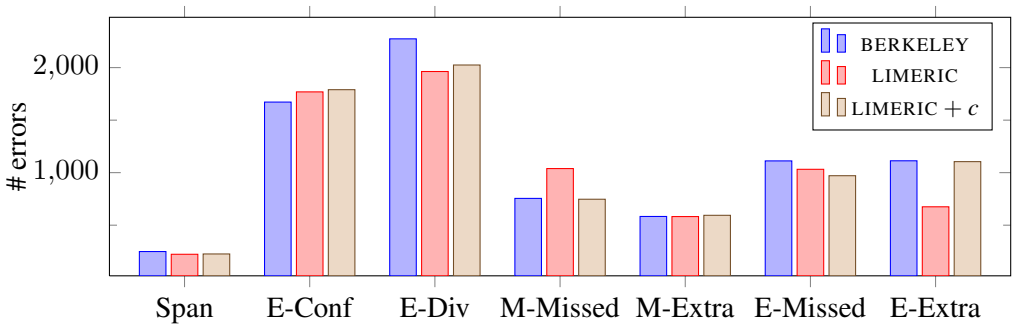

Figure 4: error counts in fine grained categories of Kummerfeld and Klein (2013)

6 System analysis

Features

Since we use simple, linear learning, it is possible to analyse feature weights to introspect system performance. In particular, we would like to understand why our stack competition features, which are well principled, did not give a substantial performance gain. We do this by analysing the number of non-zero

[6] https://code.google.com/p/berkeley-coreference-analyser/

Features	% non-zero	avg. mag.
Surface	17.4	0.23
POS	61.9	4.55
Shape	57.3	3.77
Str match	93.7	20.60
Length	48.7	2.37
Distance	92.5	19.74

Features	% non-zero	avg. mag.
Attr agree	88.3	24.88
Attr discord	74.6	20.00
WN similarity	94.5	21.67
Competition	**87.9**	**15.28**
Likelihood	**61.4**	**7.47**
Depth	**93.9**	**11.73**

Table 2: proportion of features within a set with non-zero weight in LIMERIC+*c* model (left) and average magnitude of this weight across the set (right); novel features are indicated in bold

features in our feature sets as an indication of how often they were useful in distinguishing predictions, and average feature weight magnitude as an indication of how trusted they were in inference.

Given the performance decrease of LIMERIC+*c* against our base system, it is surprising that it is the competition features which appear to be the best performing of our novel feature space. We are cautious that their very high feature weight could represent overfitting and future work could use regularisation, as well as explore any discourse level differences between TRAIN and the TEST datasets.

Depth in stack performs well, particularly given that it captures similar information to Distance and feature weight needs to be shared between the two feature sets. The least useful feature set is Surface, probably due to our large feature space size and its sparseness. Since this comprises the greatest number of features, we anticipate its deactivation will improve efficiency for a minimal impact on performance.

Stack

Our reported performance is based on a search of the full stack, but this gives rise to a large time cost which is not practical given the role of coreference resolution to inform downstream inference. While recency is important cue for coreference, it is not clear what bounds we can place on candidate generation while maintaining good performance. Figure 5 plots the depth from the top of the stack of the correct reduce operation in DEV.

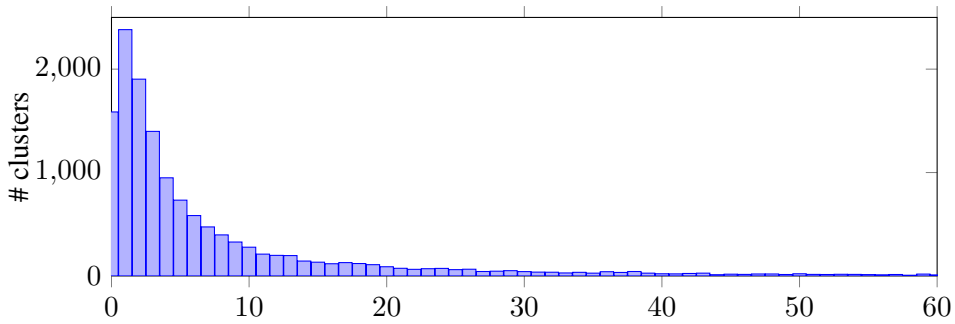

Figure 5: distribution of correct merge targets in the stack, DEV

We note a very long tail to this distribution and cut it off at depth 60, which cumulatively represents 97% of the data. The vast majority of correct merge targets are near the top of the stack, with 78% up to depth 10 and 88% up to depth 20. Setting maximum search depth to 60 yields a model which scores 61.31 on DEV. While this outperforms Stanford and is competitive with Berkeley, the magnitude of loss is surprising given the distribution in Figure 5. Error analysis shows an increased number of Conflated Entity and Extra Mention errors, which were shown in Kummerfeld and Klein (2013) to have a substantial precision cost. We note that this is consistent with our system having good accuracy in predicting whether or not a new entity cluster should form, but being restricted to choose an incorrect merge target when the correct one is outside its search window.

Configuration	MUC	B^3	CEAFE	CoNLL
LIMERIC+*c*	**70.67**	**68.33**	50.55	**63.18**
classifier scoring	70.36	68.12	50.65	63.04
# beams=1	70.52	68.21	**50.66**	63.13
no beam threshold	69.60	67.42	50.12	62.38

Table 3: impact of various parameters for beam search, DEV

Beam search

Beam search affects both time and space complexity since each classification step proposes new stacks which need to be compared for pruning. Our final system uses a maximum beam size of 10 with a conservative threshold of 5 for new stack formation. We find little difference between using classification score or stack discourse likelihood as our pruning metric. The results in Table 3 indicate that beam search isn't essential for state of the art performance in our system, our rich feature set is adequate alone. If we limit the beam to a single stack, we still have competitive performance with CL^3M. Indeed, if we do not set a strict threshold on the score at which a new stack is formed, we are forced to maintain the maximum 10 stacks and this actually hurts performance. These findings are consistent with those in Zhang and Nivre (2012), which demonstrates that performance gains are only seen from beam search at run time when their shift-reduce parser was trained similarly, maintaining a beam of potentially incorrect predictions and learning to recover as well as possible from unfavourable states.

7 Conclusion

The primary contribution of our work is a reconceptualisation of coreference by analogy with the shift-reduce parsing algorithm. We present LIMERIC, a simple, low memory coreference resolution system which achieves the competitive CoNLL scores of 63.33 and 62.91 on the CoNLL-2012 DEV and TEST splits of OntoNotes 5. Our framework unites ranking- and cluster-based approximations to global consistency encoding, and we outperform systems using either in isolation. By interpreting the stack of incrementally growing entity clusters in our system as a reader's mental status while reading, we naturally extend the shift-reduce algorithm to express a series of rich discourse features which perform well in feature analysis. Our results demonstrate the promise of psycholinguistic insights for coreference resolution and future directions include further extension of our discourse, as well as semantic, model. We plan future work in enriching our training process with beam search, and incorporating more insights from Centering and Accessibility Theories.

8 Acknowledgements

The authors thank the anonymous reviewers for their helpful feedback, and Will Radford, Joel Nothman, and Matthew Honnibal for their contribution to this work. The first author was supported by an Australian Postgraduate Award scholarship. This work was supported by ARC Discovery grant DP1097291 and the Capital Markets CRC Computable News project.

References

A. V. Aho and S. C. Johnson. 1974. LR parsing. *ACM Computing Surveys*, 6(2):99–124.

Mira Ariel. 2001. Accessibility theory: An overview. *Text representation: Linguistic and psycholinguistic aspects*, pages 29–87.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *The first international conference on language resources and evaluation workshop on linguistics coreference*, volume 1, pages 563–566, Grenada, Spain.

Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 294–303, Edinburgh, Scotland.

Kai-Wei Chang, Rajhans Samdani, and Dan Roth. 2013. A constrained latent variable model for coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 601–612, Seattle, Washington.

Pascal Denis, Jason Baldridge, et al. 2007. Joint determination of anaphoricity and coreference resolution using integer programming. In *Proceedings of NAACL HLT*, pages 236–243, Rochester, New York.

Greg Durrett and Dan Klein. 2013. Easy victories and uphill battles in coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1971–1982, Seattle, Washington.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, Massachusetts.

Eraldo Fernandes, Cícero dos Santos, and Ruy Milidiú. 2012. Latent structure perceptron with feature induction for unrestricted coreference resolution. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 41–48, Jeju Island, Korea.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2007. English gigaword (third edition), LDC catalog number LDC2003T05.

Barbara J. Grosz, Scott Weinstein, and Aravind K. Joshi. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21:203–225.

Manfred Klenner and Don Tuggener. 2011. An incremental entity-mention model for coreference resolution with restrictive antecedent accessibility. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 178–185, Hissar, Bulgaria.

Jonathan K Kummerfeld and Dan Klein. 2013. Error-driven analysis of challenges in coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 265–277, Seattle, Washington.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 28–34, Portland, Oregon.

Thomas Lin, Oren Etzioni, et al. 2012. No noun phrase left behind: detecting and typing unlinkable entities. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 893–903, Seattle, Washington.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 25–32, Vancouver, British Columbia.

Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 104–111, Philadelphia, Pennsylvania.

Simone Paolo Ponzetto and Michael Strube. 2006. Exploiting semantic role labeling, wordnet and wikipedia for coreference resolution. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 192–199, New York, New York.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages 1–40, Jeju Island, Korea.

Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nate Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. 2010. A multi-pass sieve for coreference resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 492–501, Cambridge, Massachusetts.

Altaf Rahman and Vincent Ng. 2009. Supervised models for coreference resolution. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 968–977, Edinburgh, Scotland.

Marta Recasens, Marie-Catherine de Marneffe, and Christopher Potts. 2013. The life and death of discourse entities: Identifying singleton mentions. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 627–633, Atlanta, Georgia.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational linguistics*, 27(4):521–544.

Veselin Stoyanov, Claire Cardie, Nathan Gilbert, Ellen Riloff, David Buttler, and David Hysom. 2010a. Coreference resolution with reconcile. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 156–161, Uppsala, Sweden.

Veselin Stoyanov, Claire Cardie, Nathan Gilbert, Ellen Riloff, David Buttler, and David Hysom. 2010b. Reconcile: A coreference resolution research platform.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the 6th conference on Message understanding*, pages 45–52, Columbia, Maryland.

Yue Zhang and Joakim Nivre. 2012. Analyzing the effect of global learning and beam-search on transition-based dependency parsing. In *Proceedings of COLING 2012: Posters*, pages 1391–1400, Bombay, India.