Labeling the Semantic Roles of Commas*

Naveen Arivazhagan and **Christos Christodoulopoulos** and **Dan Roth**
Department of Computer Science
University of Illinois at Urbana-Champaign
{arivazh2, christod, danr}@illinois.edu

Abstract

Commas and the surrounding sentence structure often express relations that are essential to understanding the meaning of the sentence. This paper proposes a set of relations commas participate in, expanding on previous work in this area, and develops a new dataset annotated with this set of labels. We identify features that are important to achieve a good performance on comma labeling and then develop a machine learning method that achieves high accuracy on identifying comma relations, improving over previous work. Finally, we discuss a variety of possible uses, both as syntactic and discourse-oriented features and constraints for downstream tasks.

1 Introduction

Understanding the relations expressed in text is important to many practical tasks such as question answering, information extraction, and textual entailment. A key element of written text that is used to connect ideas is the comma. It has been described as "the most ubiquitous, elusive and discretionary of all stops" (Jarvie 1995, p. 10). While the comma is on par with the period in that both account for $\sim$45% of all marks in the Brown corpus (Francis and Kucera 1982), (Meyer 1987, p. 24–25) observes that "[t]he comma is the most versatile mark of punctuation".

Commas often mark a relation between the segments of text they connect. This can be used as a cue to extract relations or information from the corresponding segments. While it has been argued (Meyer 1987), that the semantics of commas relate only to their function (and hence limited to being either *delimiter* or *separator commas*), we view the commas themselves as explicit markers of semantic relations between the segments they are separating or delimiting. For example, in the sentence, '*We invited the computer scientists , Susan and Hanna*', knowing that the comma marks a SUBSTITUTE, rather than a LIST relation, means that we can tell that Susan and Hanna are the scientists and not just two random guests.

*This material is based on research sponsored by DARPA under agreement number FA8750-13-2-0008. This research is also supported by NIH grant R01-HD054448-07

In this paper we build on the corpus previously annotated by (Srikumar et al. 2008) by refining existing relations and adding new ones. While the original corpus had extracted relations for only 57% of the commas (with every other comma having an OTHER label), with the new relations we add, we are able to label the relations that 99% of the commas entail. The new data set can be used to learn the new relations that have been identified. We also show that certain modeling assumptions made in previous work, specifically creating a mapping between the local constituency parse context of a comma to the comma role (Bayraktar, Say, and Akman 1998), are inherently ambiguous and do not hold as well as expected.

Finally, we build a simple model to learn these new relations and outperform previous systems. In doing so, we have identified a set of features that are important to consider when addressing the problem of semantic comma role labeling.

2 Related Work

Previous work in natural language processing has recognized the semantic importance of commas and made progress in classifying them according to their roles. (Bayraktar, Say, and Akman 1998) analyzed the WSJ portion of the Penn Treebank (PTB; (Marcus, Santorini, and Marcinkiewicz 1993)), and identified a set of seven roles that commas can have in a sentence. They defined a *comma-syntax-pattern* to be the syntactic parent of a comma and all the siblings of a comma, in the parse tree of the sentence. They postulated that there exists a function from these comma-syntax-patterns to the roles of the commas and created a mapping from 211 comma-syntax-patterns to comma roles by analyzing a candidate sentence for each comma-syntax-pattern. This gave them an 80% coverage of all the commas in the annotated portion of the WSJ, however they did not test their approach against a set of manually annotated commas.

Also using the WSJ corpus, (Srikumar et al. 2008) label commas with their semantic roles and extract entailed relations. They do this by mining for templates of the entailed relations in the parse trees of these sentences which can then be matched to the parse trees of new sentences to classify the comma and extract relations.

The problem of Chinese sentence segmentation has been treated as a comma classification problem (Xue and Yang 2011). They formulate a binary classification problem where they try to classify commas into end-of-sentence and not-end-of-sentence. While they are classifying commas according to their syntactic roles, their work is similar to ours in terms of the features used.

There are a number of works in linguistics that focus on the syntactic, semantic and pragmatic role of punctuation (and of the comma in particular). (Meyer 1987) provides an extensive account of commas and their uses, in syntactic, semantic and discourse context. However, while his study is descriptive, with plenty of examples, he does not try to formalise any heuristics for identifying the different uses of punctuation.

(Dale 1991) presents the view that punctuation, along with lexical and graphical markers, also plays the role of a discourse marker. He observes, however, that punctuation often under-determines specific discourse relations but can be used to extract discourse structure.

3 A Corpus of Comma Relations[1]

3.1 Label Description

We build on the corpus previously annotated by (Srikumar et al. 2008) by refining existing relations and adding new ones. The original corpus was built by selecting 1,000 sentences, all containing at least one comma, from section 00 of the WSJ portion of the Penn Treebank. These sentences were manually annotated with four types of relations:

SUBSTITUTE indicates an apposition structure:

(1) *Mary , the machine-learning scientist , was having a nice day.*

ATTRIBUTE describes commas that separate a noun from a non-restrictive/non-essential modifier:

(2) *Her friend Susan , also a scientist , was coming back from work.*

LOCATIVE commas specify a *located-in* relation such as city or state:

(3) *She worked at Google , New York.*

LIST is used to label commas that separate elements in a list:

(4) *She was carrying her bag , her umbrella , and her phone.*

Finally, OTHER is used to label commas that did not indicate any of the above four relations, however, it accounts for more than 43% of the total commas (Table 1).

In their work, (Bayraktar, Say, and Akman 1998) used a label set which included the above labels (excluding OTHER and LOCATIVE) and also four new labels, INTRODUCTORY, COMPLEMENTARY, INTERRUPTER, and QUOTATION. These can be viewed as a refinement of the OTHER category in the annotations of (Srikumar et al. 2008). They arrived at this label set by studying previous work on the different classes of uses of commas. Since there was no consensus on the number of different classes of the use of commas, they decided to work with the one of the popular style guides at the time, (Ehrlich 1992), and arrived at the above set of seven labels. Here, we provide a brief definition of each of these new labels.

INTRODUCTORY is used to separate an introductory element at the start of the sentence from the main clause:

(5) *Without thinking , Mary ran down the street.*

COMPLEMENTARY is used to separate a complementary element from the main clause:

(6) *She ran down the street , yelling at Susan to stop.*

INTERRUPTERs are used to label commas that delimit an interrupter. An interrupter is a word, phrase, or clause that that occurs in the middle of a sentence, breaking its logical flow:

(7) *Susan , finally , stopped.*

QUOTATION is used to label commas that set off direct speech made by a speaker:

(8) *"Sorry, I couldn't hear you" , she said.*

Thus we have a set of 9 labels, the seven labels from (Bayraktar, Say, and Akman 1998), LOCATIVE, and OTHER. This new OTHER category contains commas that do not fall into any of the other 8 categories. They mark less interesting or relatively rare phenomena, for example, being used for typographical reasons such as *date, year* (July 27, 2005) or as thousandths markers (10,000) or to demarcate the the boundaries of a long NP.

3.2 Annotation

We undertake two annotation tasks. The first is to produce the comma-syntax-pattern to label mappings as was done by (Bayraktar, Say, and Akman 1998). This will serve as our baseline. We identified all the comma-syntax-patterns from the WSJ section of PTB (approximately 2,000 unique patterns). To simplify the annotation task, we selected only the most frequent comma-syntax-patterns and proceeded to annotate this list of comma-syntax-patterns by observing candidate parses and the role the comma plays in the sentence. We labeled 159 of the most frequently appearing comma-syntax-pattern. This gave us a 76% coverage of all the commas in PTB. Since the available comma-syntax-pattern annotations only cover 76% of all the comma-syntax-patterns, we were only able to re-annotate 60% of the OTHER labels. Note that the 60% coverage of OTHER is not because none of the remaining commas fall into one of the new categories, rather, it is simply because we do not map all the comma-syntax-patterns we come across to a specific category.

In the second annotation task we manually refine all the OTHER labels in the corpus of (Srikumar et al. 2008) so as to reflect the new labels. The definitions of SUBSTITUTE, ATTRIBUTE, and LIST used by (Srikumar et al. 2008) have an exact one-to-one correspondence with three labels from (Bayraktar, Say, and Akman 1998), so we do not need to relabel commas with those labels. Similarly, the LOCATIVE label has no overlap with any of the four new labels so they

[1]The dataset is available at:
http://cogcomp.cs.illinois.edu/page/publication_view/780

Rel type	#	Agreement
ATTRIBUTE	288	0.81
LIST	328	0.69
LOCATIVE	98	0.93
SUBSTITUTE	379	0.80
OTHER	835	0.95
↳ COMPLEMENTARY	170	0.78
↳ INTERRUPTER	211	0.89
↳ INTRODUCTORY	324	0.89
↳ QUOTATION	108	0.80
↳ OTHER	22	0.50
TOTAL	1928	

Table 1: Distribution of comma labels in (Srikumar et al. 2008)'s corpus and the refined OTHER labels. Agreement on the refined OTHER labels is calculated using Cohens's kappa statistic. The annotator agreement statistics on the old labels, as measured by accuracy, are taken from (Srikumar et al. 2008)

do not need to be re-annotated either. Two annotators refined the same 34% (305/892) of the OTHER commas into the new set of 9 labels. The remaining was annotated by one of the two annotators. The annotators followed the definitions and guidelines provided in (Bayraktar, Say, and Akman 1998), with minor adjustments to align our decisions with those of (Srikumar et al. 2008), since some of the OTHER labels in their corpus were re-annotated with one of the four original labels. The distribution of labels resulting from the manual refinement, and Cohens's kappa coefficient for annotator agreement are shown in Table 1. The annotator agreement statistics on the old labels, as measured by accuracy, are taken from (Srikumar et al. 2008). Since they had four annotators, accuracy closely resembles the kappa score as the chance agreement of the annotators would be very near zero. For the rest of our experiments we consider the annotations by the annotator who annotated all the OTHER commas to be the gold standard.

4 Examples of Use in Downstream Applications

As (Meyer 1987, p.9) notes, the function of commas "is not [. . .] simply syntactic, prosodic, or semantic. It involves the complex interaction of all of these factors." The labels we describe above capture most of the information that the comma provides, across syntax and semantics. Here, we will demonstrate how that information can be used in various NLP applications, either as features or as constraints.

Starting with their most basic functionality as indicators of coordinated clauses/phrases, commas labels can be used during syntactic parsing (or parse reranking (Charniak and Johnson 2005)). By distinguishing between delimiting and separating types of commas, we can enforce the creation of distinct or continuous parse constituents.

SUBSTITUTE and LOCATIVE commas will, in all likelihood, participate in Noun Phrase constituents, whereas LIST commas have to unify phrases of the same type and result in a parent constituent of that type.

On the other hand, INTRODUCTORY, COMPLEMENTARY, INTERRUPTER and QUOTATION, introduce constituents that have to have a distinct phrase-structure label and need to be siblings of their modifying phrases. Of course, given the fact that our comma classifier is dependent on syntactic parse features (see section 5.1), it might be worth considering a joint learning, or joint inference (Roth and Yih 2004) approach. Previous works have already shown that semantic role labeling can benefit from such joint approaches. For example, (Toutanova, Haghighi, and Manning 2005) have shown that performance of semantic role labeling systems can be improved by jointly classifying the arguments using re-ranking based approaches. (Srikumar and Roth 2011), using joint inference under the constrained conditional modeling framework (Chang, Ratinov, and Roth 2012), have shown how combining semantic role labeling for for different tasks (verbs and prepositions) can improve performance.

Beyond their syntactic interpretation, commas can be used in semantic (Tang and Mooney 2001) and discourse parsing (Lin, Ng, and Kan 2012). As mentioned in the Introduction, knowing that a comma marks a SUBSTITUTE, rather than a LIST relation helps us identify the unique entities. It is easy to show that labels such as QUOTATION can be used directly as indicators of attribution (phrase attributed to a specific speaker in discourse), whereas e.g. INTRODUCTORY can indicate implicit discourse markers.

Another use of comma labels is in determining whether some information should be included in our analysis of the text or not. For instance in the following example

(9) *It was Mary , not Susan , who was yelling.*

if the commas are labeled as INTERRUPTER, we could ignore the extra piece of information that tells us that Susan was not yelling, whereas if we used the LIST label, that information would have equal weight as the main proposition (i.e. *Mary was yelling* $\wedge$ *Susan was not yelling*).

Given the discourse parsing task, some of the labels used here could be potentially refined further. For example, taking inspiration from the Penn Discourse Treebank (Prasad et al. 2008) we could have labels such as the following.
CONSEQUENCE

(10) *Having ran for an hour , Jane crashed.*

CONTINUATION

(11) *She looked for a snack , while continuing to recover.*

or TEMPORAL

(12) *Opening the fridge , she found an apple.*

However, since these refinements belong exclusively to the task of discourse parsing, we believe that they would diminish some of the generality of our approach. (Dale 1991) suggests that this might be a real limitation of punctuation marks in general. He argues that "at least some of the punctuation markers are so lacking in specificity that they tell us not very much at all about the particular relation they realize". The question of whether this is true of commas is beyond the scope of this paper. Nonetheless, we chose to use the broader set of labels which attempt to capture all the information that the comma structure supplies and delegate the task of further refinement —if needed— to the downstream application.

5 Building a Comma Classifier

We wanted to create a comma role classification system for two reasons: as a verification of our annotated corpus, and also in order to provide a proof-of-concept tool for further use in downstream NLP applications.

We use a combination of syntactic and semantic features train a classifier that can predict the comma roles.

5.1 Syntactic Features

We use part-of-speech tags (POS) from the Illinois POS tagger (Even-Zohar and Roth 2001), shallow parses from the Illinois Shallow Parser (Punyakanok and Roth 2001), and constituency parses from the Charniak parser (Charniak and Johnson 2005). The syntactic features that we use are (Features for the first comma in Figure 1 have been provided as an example. They are represented as $x : y : value$ where x is the size of the ngram, y is the index of the ngram in the window, and $value$ is the value of the feature):

Word The words from the sentence in a window of size 4 centered around the comma: *1:1:George*, *1:2:Crenshaw*, *1:3:Dean*, *1:4:of*

POS (Part of Speech) The 2grams, 3grams, 4grams and 5grams of part of speech tags of words in a window of size 10 centered on the comma (The POS tag for determiners is split if the word is *the*): *2:4:NNP-NNP*, *2:5:NNP-NNP*, *2:6:NNP-IN* ...

Chunk (Shallow Parse) unigrams and bigrams of the chunk labels obtained from a shallow parse in a window of size 4 around the comma : *1:1:null*, *1:2:NP*, *1:3:NP*, *1:4:PP* , *2:1:null-NP*, *2:2:NP-NP*, *2:3:NP-PP*

Parse (Constituency Parse) The ungirams and bigrams of the constituent labels of the adjacent siblings of the comma and its parent in the constituency parse. The labels are augmented with the label of the first child of the corresponding constituent in the parse tree. A window of size 4 around the comma and 5 around the comma parent is used: *s2:1:null-NP+NNP*, *s2:2:NP+NNP-NP+NP*, *s2:3:NP+NP-comma*, *p1:3:NP+NP*, *p2:3:NP+NP-VP+MD*

Bayraktar-Label (Constituency Parse) The label of the comma-syntax-pattern as obtained from the constituency parse: SUBSTITUTE

5.2 Semantic Features

While the syntactic features completely capture all elements of the comma structure, we find, in our experiments, that they are in fact not sufficiently discriminative by themselves. This ambiguity is however resolved if we add some semantics as to what the segments mean. This can be done with

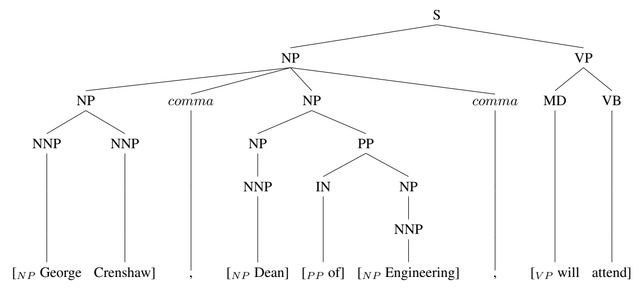

Figure 1: A sample sentence containing POS, constituency parse, and shallow parse annotations used as features.

a named entity recognition system. Consider the example: *Chicago, Illinois, is a windy city.* This sentence has exactly the same syntactic structure as the sentence in Figure 1. However, the first comma in this sentence is not a SUBSTITUTE comma, but rather a LOCATIVE comma. If we realize that *George Crenshaw* is a person and that *Chicago* and *Illinois* are locations, then we can easily disambiguate between a LOCATIVE and a SUBSTITUTE comma. We used the Illinois Named Entity Tagger (Ratinov and Roth 2009) to obtain named entity (NE) tags. Note that the NE feature is added only if the size of the constituent is no greater than 1.5 times the size of the named entity segment. The size restriction is added because we are only concerned about whether the whole phrase is a certain type of entity and not about whether it simply contains one.

Another source of features could be Semantic Role Labeling (Palmer, Gildea, and Xue 2010). For example, if the span of text to a certain side of the comma is an argument or an argument modifier of a verb or noun on the other side of the comma, then we get a clue as to what relation is crossing over the comma. In the sentence *At Cray Computer, he will be paid $24,000.*, knowing that *At Cray Computer* is an argument modifier (AM-LOC) of the verb *paid* tells us that that the first portion of the sentence modifies the main portion of the sentence. This is the function of an INTRODUCTORY comma.

6 Learning Problem Formulation

Given a sentence s and a sequence of commas specified as indices in the sentence, $\mathbf{c} = (c_1, ..., c_n)$, the goal is to produce a sequence of output labels $\mathbf{y} = (y_1, ..., y_n)$, where $y \in Y = \{$ATTRIBUTE, COMPLEMENTARY, INTERRUPTER, INTRODUCTORY, LIST, LOCATIVE, QUOTATION, SUBSTITUTE, OTHER$\}$, corresponding to the commas. This is a sequence labeling problem.

We found, however, that commas are most often context free and only play a role within the constituent in which they are found. We trained two sequential models 1) over all the commas in a sentence, 2) over only commas that were siblings in the parse. In comparison to the results in Table 3, these gave us an accuracy of 82.1% and 86.2% respectively on **Gold-Gold** and 78.9% and 79.8% on **Auto-Auto**. The lower performance of the structured classifier as compared to the local classifier is due to overfitting as evidenced by the

Setting	Accuracy
Gold-Gold	87.1
Auto-Auto	83.6
Gold-Auto	81.9
Bayraktar-Gold	85.3
Bayraktar-Auto	80.5

Table 2: Performance of learned classifier under different settings over all 9 labels compared to the baseline approach of (Bayraktar, Say, and Akman 1998).

higher performance of the structured classifier (99.8% accuracy) than the local classified (98.8% accuracy) on the train set itself.

Due to the observation that comma labels are independent of other commas outside of the local context – the commas parent constituent – we are able to instead model it as a local classification problem with sufficiently rich features that capture this relevant context. So we instead use training examples of the form $(< s, c >, y)$. We build this classifier by training on a combination of the corpus and comma-syntax-pattern annotations we produced, using a Sparse Averaged Perceptron (Jackson and Craven 1996) with LBJava (Rizzolo and Roth 2010) trained over 160 rounds (learning rate = 0.024, thickness = 3.9).

7 Experiments & Results

7.1 Baseline - Comma-Syntax-Pattern

As a baseline we use the approach of (Bayraktar, Say, and Akman 1998). We use the comma-syntax-pattern annotations that were produced to make predictions about the label of a comma from the manually annotated corpus. Only those commas whose comma-syntax-pattern annotations are available are evaluated under the baseline. A prediction is considered to be correct if and only if it matches the comma label in the manually annotated which we consider to be the gold standard. In this way we evaluate the baseline using the comma-syntax-patterns derived from the gold and automatically induced parses. The results are reported in table 2.

One of the reasons for the errors in the approach of (Bayraktar, Say, and Akman 1998) is that it does not address the potential ambiguity in the mapping from comma-syntax-patterns to labels. Examples (13) and (14), both have the same comma-syntax-pattern (*NP → NP , NP ,*). However, their labels are different – LOCATIVE and SUBSTITUTE, respectively.

(13) *Chicago, Illinois, is a windy city.*

(14) *George Crenshaw, Dean of Engineering, will attend*

Another problem is that comma-syntax-patterns do not accommodate the possibility of commas siblings in the parse tree having different labels. For example, in the sentence

(15) *Shearson Inc. , New York , which is 62%-owned by Houston Co. , consented to a fine .*

Label	Precision	Recall	F1
ATTRIBUTE	78.8	79.9	79.3
LIST	87.9	83.8	85.8
LOCATIVE	92.2	84.7	88.3
SUBSTITUTE	86.3	86.3	86.3
OTHER			
↳ COMPLEMENTARY	76.0	80.0	77.9
↳ INTERRUPTER	72.2	73.9	73.1
↳ INTRODUCTORY	87.8	93.5	90.6
↳ QUOTATION	89.2	91.7	90.4
↳ OTHER	100.0	13.6	24.0
Accuracy		**83.6**	

Table 3: Comma Role Labeling performance of learned classifier under **Auto-Auto** using the new label set.

the first two commas which are siblings have the same comma-syntax-pattern (*NP → NP , NP , SBAR*). However, the labels of the commas are SUBSTITUTE and ATTRIBUTE respectively.

Despite the above two potential problems the baseline performs surprisingly well. However, its coverage is still lacking. Since not all comma-syntax-patterns have been annotated with the labels that they should map to, only 77.9% of the manually annotated commas could be labeled using this process.

7.2 Learned Classifier

We evaluate our classifier under three different settings:

- **Gold-Gold**: train and test on gold syntactic features (POS, parse trees). We strip the functional tags off the gold parse trees during training, as they are not included by the Charniak parser.
- **Gold-Auto**: train on gold, test on automatically induced features
- **Auto-Auto**: train and test on automatically induced features

In all experiments, 5-fold cross-validation is used to evaluate the performance of the learned classifier specified in the previous subsection. All reported results were obtained with a confidence of 95% on an interval of +/-1%. Table 3 shows the results for each label under the **Gold-Gold** and **Auto-Auto** settings.

In the **Auto-Auto** setting, our classifier obtains an accuracy of 83.6%. Thus we achieve a significant error reduction of 15.9% over the baseline approach of (Bayraktar, Say, and Akman 1998) as evaluated on automatically induced parses. The main advantage of our approach over the baseline is that we are able to predict the labels of all the commas and are not constrained by the availability of their comma-syntax-pattern annotations. Thus we have 100% coverage as compared to the 77.9% coverage obtained by the approach of (Bayraktar, Say, and Akman 1998), which, assuming Zipff's law holds true for syntax patterns, would have needed a significant annotation effort to be comparable. Another benefit

Features	**Gold-Gold**	**Auto-Auto**
All	87.1	83.6
- NER	87.0	83.4
- POS	86.1	82.3
- Constituency Parse	74.1	74.1
- Shallow Parse	86.7	83.2

Table 4: Ablation analysis of the classifier for on **Gold-Gold** and **Auto-Auto**. The ablation is performed based on the annotations required to produce a group of features.

of the learned classifier over the baseline is that its accuracy would grow with an increase in annotated data. On the other hand, there is little evidence to suggests that the baseline performance would grow even if we annotated more comma-syntax-patterns.

We were not able to make a comparison of the comma classification performance between our method and that of (Srikumar et al. 2008) because they were trying to solve the joint problem of extracting relations and labeling commas. However, on the OTHER comma, for which they were also only classifying the comma, they obtained an F1 score of 85.2 in the **Gold-Gold** setting. Our method is able to obtain an F1 score of 89.2 in the **Gold-Gold** setting when evaluated on the old label set. Our method is also more versatile than the template mining approach in that it can be expanded to use new features quite easily.

Ablation study As seen in the ablation analysis (Table 4), the most important features are those that are derived from the constituent parse followed by POS features. The reason for the large contribution to the classifier by syntactic features is the nature of the label set. As discussed in section 4, the labels were chosen finely enough to capture all the information that commas supply, but also broadly enough so as to not interfere with the predictions of downstream tasks such as discourse parsing. We find that at this level of refinement the syntactic features are able to disambiguate commas very well by themselves. The NE feature improves the F1 score of the LOCATIVE tags by 0.2. It primarily helps in identifying locations and thus disambiguating from other comma labels. To a smaller extent it also improves the F1 score of SUBSTITUTE and QUOTATION by identifying people and organizations that take part in apposition relations in the former, and by identifying people saying things in the latter.

Error analysis A number of errors are caused due to not properly handling the possibility that a comma can take on more than just one role. In the following sentence, the comma could take on either role: SUBSTITUTE, or COMPLEMENTARY, but the system is penalized for choosing one instead of the other.

(16) *Sales in October rose 18 % from a year earlier to 500,004 units , a record for the month **,** the Japan Automobile Dealers ' Association said .*

Most errors on ATTRIBUTE commas are very often due to them being conflated with the other roles thus causing ATTRIBUTE to have the least recall. This is due to a number of reasons. The difference between the ATTRIBUTE commas and the SUBSTITUTE commas is the that post-modifier of the noun is non-restrictive in the former and restrictive in the latter. Due to this, the classifier makes errors between the two when the provided features are unable to distinguish a restrictive or non-restrictive phrase. In the following sentence the classifier is not able to recognize that the noun phrase: *most purchased at nominal prices* is a non-restrictive phrase.

(17) *The management group owns about 18 % of the stock **,** most purchased at nominal prices , and would stand to gain millions of dollars if the company were sold .*

Additionally, both the ATTRIBUTE label and the INTRODUCTORY and COMPLEMENTARY labels can also act as modifiers and the only distinguishing factor is that ATTRIBUTE commas modify nouns whereas the other two can modify or describe general events. In examples (18) and (19), the ATTRIBUTE commas are mislabeled as COMPLEMENTARY and INTRODUCTORY respectively.[2]

(18) *Failure to complete the form had been punishable as a misdemeanor until last November **,** when Congress determined that the crime was a felony punishable by up to 10 years in prison .*

(19) *Founded by Brooklyn , N.Y. , publishing entrepreneur Patricia Poore **,** Garbage made its debut this fall with the promise to give consumers the straight scoop on the U.S. waste crisis .*

(20) *Many banks **,** particularly smaller ones , were slow to computerize and couldn't target market niches that would have made the programs more profitable*

Finally, some of the reported errors are actually due to incorrect annotations: the classifier predicts the correct label but the provided initial annotation it is being compared against is incorrect. For example, the comma in sentence (20) was incorrectly annotated as INTERRUPTER, but the classifier correctly predicts ATTRIBUTE. These errors will be addressed in the next version of the corpus.

8 Conclusion

We expanded the original Comma Resolution Task proposed by (Srikumar et al. 2008) to include four new labels originally suggested in (Bayraktar, Say, and Akman 1998). This allows us to label the relation induced by almost 99% of all commas as compared to the 57% coverage obtained by (Srikumar et al. 2008). We also make suggestions on how these comma labels could benefit downstream tasks. We identified features that are important to comma labeling and finally built a model that significantly improves on both the performance and the coverage of previous systems.

[2]A dependency parser could have helped address these kinds of errors. However, for these instances, the parser produced incorrect dependencies. Like most other automatically induced features, dependencies do resolve some of the problematic cases, but tend to create new ones.

References

Bayraktar, M.; Say, B.; and Akman, V. 1998. An analysis of english punctuation: The special case of comma. *International Journal of Corpus Linguistics* 3(1):33–57.

Chang, M.; Ratinov, L.; and Roth, D. 2012. Structured learning with constrained conditional models. *Machine Learning Journal*.

Charniak, E., and Johnson, M. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 173–180. Ann Arbor, Michigan: ACL.

Dale, R. 1991. The role of punctuation in discourse structure. In *Working Notes for the AAAI Fall Symposium on Discourse Structure in Natural Language Understanding and Generation*, 13–14.

Ehrlich, E. 1992. *Schaum's Outline of Punctuation, Capitalization & Spelling*. McGraw-Hill Education.

Even-Zohar, Y., and Roth, D. 2001. A sequential model for multi class classification. In *EMNLP*, 10–19.

Jackson, J. C., and Craven, M. W. 1996. Learning sparse perceptrons. *Advances in Neural Information Processing Systems* 654–660.

Jarvie, G. 1995. *Chambers Good Punctuation Guide*. Chambers.

Lin, Z.; Ng, H. T.; and Kan, M.-Y. 2012. A PDTB-styled end-to-end discourse parser. *Natural Language Engineering* 20(02):151–184.

Marcus, M. P.; Santorini, B.; and Marcinkiewicz, M. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics* 19(2):313–330.

Meyer, C. F. 1987. *A linguistic study of American punctuation*. New York: Peter Lang.

Palmer, M.; Gildea, D.; and Xue, N. 2010. Semantic role labeling. *Synthesis Lectures on Human Language Technologies* 3(1):1–103.

Prasad, R.; Dinesh, N.; Lee, A.; Miltsakaki, E.; Robaldo, L.; Joshi, A. K.; and Webber, B. L. 2008. The Penn Discourse TreeBank 2.0. In *LREC*.

Punyakanok, V., and Roth, D. 2001. The use of classifiers in sequential inference. In *NIPS*, 995–1001. MIT Press.

Ratinov, L., and Roth, D. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of CoNLL-09*, 147–155.

Rizzolo, N., and Roth, D. 2010. Learning Based Java for rapid development of NLP systems. In *LREC*.

Roth, D., and Yih, W. 2004. A linear programming formulation for global inference in natural language tasks. In Ng, H. T., and Riloff, E., eds., *CoNLL*, 1–8. Association for Computational Linguistics.

Srikumar, V., and Roth, D. 2011. A joint model for extended semantic role labeling. In *EMNLP*.

Srikumar, V.; Reichart, R.; Sammons, M.; Rappoport, A.; and Roth, D. 2008. Extraction of entailed semantic relations through syntax-based comma resolution. In *ACL*. Columbus, OH, USA: Association for Computational Linguistics.

Tang, L. R., and Mooney, R. J. 2001. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Proceedings of the European Conference on Machine Learning (ECML)*.

Toutanova, K.; Haghighi, A.; and Manning, C. D. 2005. Joint learning improves semantic role labeling. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Xue, N., and Yang, Y. 2011. Chinese sentence segmentation as comma classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, 631–635. Association for Computational Linguistics.