SRI's Tipster II Project

Jerry R. Hobbs, Douglas Appelt, John Bear, David Israel,
Megumi Kameyama, Andrew Kehler, Mark Stickel, and Mabry Tyson
Artificial Intelligence Center
SRI International
Menlo Park, CA 94025

1 Introduction

The principal barrier to the widespread use of information extraction technology is the difficulty in defining the patterns that represent one's information requirements. Much of the work that has been done on SRI's Tipster II project has been directed at overcoming this barrier. In this paper, after some background on the basic structure of the FASTUS system, we present some of these developments. Specifically, we discuss the declarative pattern specification language FastSpec, compile-time transformations, and adapting rules from examples. In addition, we have developed the basic capabilities of FASTUS. We describe our efforts in one area—coreference resolution. We are now experimenting with the use of FASTUS in improving document retrieval and this is also described.

2 The Structure of FASTUS

FASTUS is a cascade of finite-state transducers. One can think of it as having five phases, each building up larger structures from the input. Each phase takes as its input the output objects produced by the previous phase.

1. Name Recognition
2. Basic Phrase Recognition
3. Complex Phrase Recognition
4. Clause-Level Event Recognition
5. Event Merging

In describing the system, we will say what it does, given as input the following paragraph from the management succession domain of MUC-6:

> A. C. Nielsen Co. said George Garrick, 40 years old, president of Information Resources Inc.'s London-based European Information Services operation, will become president and chief operating officer of Nielsen Marketing Research USA, a unit of Dun & Bradstreet Corp. He succeeds John H. Costello, who resigned in March.

1. The Name Recognizer recognizes the names of persons, organizations, and locations, as well as such special constructions as dates and amounts of money. There are three primary methods for this. We have patterns for recognizing the internal structure of names, as in "A.C. Nielsen Co." We have a list of common names, many of which could not otherwise be recognized, such as "IBM" and "Toys 'R' Us". Finally, we recognize or reclassify names on the basis of their immediate context. For example, if we see "XYZ's sales" or "the CEO of XYZ", then we know XYZ is a company.

In our sample text, this phase results in the following labelling:

> A. C. Nielsen Co.$_{Co}$ said George Garrick$_{Per}$, 40 years old, president of Information Resources Inc.$_{Co}$'s London$_{Loc}$-based European Information Services$_{Co}$ operation, will become president and chief operating officer of Nielsen Marketing Research USA$_{Co}$, a unit of Dun & Bradstreet Corp.$_{Co}$. He succeeds John H. Costello$_{Per}$, who resigned in March$_{Date}$.

2. The Basic Phrase Recognizer recognizes basic noun groups, that is, noun phrases up through the head noun. It also recognizes verb groups, or verbs together with their auxilliaries and embedded adverbs; certain predicate complement constructions are also analyzed as verb groups. It also labels prepositions and other particles, such as the possessive marker, relative pronouns, and conjunctions.

The core grammar for this phase is domain-independent. But there are some domain-dependent

specializations of the rules, where special semantics applies. For example, there is a general rule allowing a noun-hyphen-past participle sequence in the adjective position of noun groups, and there is a specialized version of this for a location followed by "-based", as in "London-based".

In the sample text, this phase results in the following labelling:

> [A. C. Nielsen Co.]$_{NG}$ [said]$_{VG}$ [George Garrick]$_{NG}$, [40 years old]$_{VG}$, [president]$_{NG}$ [of]$_{P}$ [Information Resources Inc.]$_{NG}$['s]$_{Poss}$ [London-based European Information Services operation]$_{NG}$, [will become]$_{VG}$ [president]$_{NG}$ [and]$_{Conj}$ [chief operating officer]$_{NG}$ [of]$_{P}$ [Nielsen Marketing Research USA]$_{NG}$, [a unit]$_{NG}$ [of]$_{P}$ [Dun & Bradstreet Corp.]$_{NG}$ [He]$_{NG}$ [succeeds]$_{NG}$ [John H. Costello]$_{NG}$, [who]$_{Relpro}$ [resigned]$_{NG}$ [in]$_{P}$ [March]$_{NG}$.

3. The Complex Phrase Recognizer recognizes complex noun groups and verb groups. For complex noun groups it attaches possessives, "of" phrases, controlled prepositional phrases, and age and other appositives to head nouns, and it recognizes some cases of noun group conjunction. For verb groups, it attaches support verbs to their content verb or nominalization complements. Some of these rules are domain-independent, but for any given domain we typically implement a number of high-priority, domain-dependent specializations of the general rules. For example, for management succession, we have complex noun groups for companies, persons, and positions. A company can have another company as a possessive, as in "Information Resources Inc.'s London-based European Information Services operation". A relational company term such as "unit" can have another company as a complement. Companies can take a company appositive. Position titles can be conjoined, and a position title can have an "of" phrase specifying the company. Persons can have position appositives.

In the sample text, this phase results in the following labelling:

> [A. C. Nielsen Co.]$_{Co}$ [said]$_{VG}$
> [George Garrick, 40 years old,
> [president of
> [[Information Resources Inc.]$_{Co}$'s
> London-based European
> Information Services
> operation]$_{Co}$]$_{Pos}$]$_{Per}$,
> [will become]$_{VG}$
> [president and chief operating officer of
> [Nielsen Marketing Research USA,
> a unit of [Dun & Bradstreet
> Corp.]$_{Co}$]$_{Co}$]$_{Pos}$
> [He]$_{Per}$ [succeeds]$_{VG}$
> [John H. Costello]$_{Per}$,
> [who]$_{Relpro}$ [resigned]$_{VG}$
> [in]$_{P}$ [March]$_{Date}$.

4. The Clause-Level Event Recognizer recognizes events in the domain of interest. This is done by matching the output of the Complex Phrase Recognizer with a set of patterns specifying the subject, verb, object, and prepositional phrases in which the events are typically expressed. In addition, locative, temporal, and epistemic adjuncts are recognized at this stage. Examples of patterns for the management succession domain are as follows:

> Person, Position
> Person becomes Position
> Person succeeds Person (as Position)
> Person resigns (from Position)

As the patterns are recognized, event structures are built up, indicating what type of event occurred and who and what the participants are. For the management succession domain, there is an event structure for a state, specifying that a person is in a position at an organization, and a structure for transitions between two states.

For the sample text, the following four event structures are constructed, corresponding to the four patterns above:

Person:	Garrick			
Position:	president			
Org:	EIS			
Person:	Garrick		Person:	Garrick
Position:	—	⇒	Position:	president
Org:	—		Org:	NMR
Person:	Costello		Person:	he
Position:	—	⇒	Position:	—
Org:	—		Org:	—
Person:	Costello		Person:	—
Position:	—	⇒	Position:	—
Org:	—		Org:	—

5. Once individual clause-level patterns have been recognized, the event structures that are built up are merged with other event structures from the

same and previous sentences. There are various constraints of consistency, compatibility, and distance that govern whether or not the two merge.

For the sample text, merging the four events found by the Clause-Level Event Recognizer results in the two following transitions, both with the same end state, the first person-centered and the second position-centered:

Person:	Garrick		Person:	Garrick
Position:	president	⇒	Position:	president
Org:	EIS		Org:	NMR
Person:	Costello		Person:	Garrick
Position:	president	⇒	Position:	president
Org:	NMR		Org:	NMR

This result is then mapped into the desired template, which may be different since in general its structure will be determined by retrieval requirements rather than how the information is typically expressed in texts.

3 FastSpec: A Declarative Specification Language

In the first version of FASTUS (Hobbs et al., 1992), the finite-state transducers were represented in a table of state changes with blocks of code associated with the final states. Only the developers were able to define patterns in this system. The next version, used in MUC-5 (Appelt et al., 1993), had a graphical interface for defining state changes and allowed blocks of code to be associated with transitions. Only a small group of cognoscenti were able to use this system.

One of the first accomplishments of the current project was the definition and development of a declarative specification language called FastSpec. It enabled the easy definition of patterns and their associated semantics, and made it possible for a larger set of users to define the patterns.

FastSpec allows the definition of multiple grammars, one for each phase. The terminal symbols in the grammar for a phase correspond to the objects produced by the previous phase, and their attributes can be accessed and checked. The rules have a syntactic part, expressing the pattern in the form of a regular expression, with attribute and other constraints permitted on the terminal symbols. They also have a semantic part, which specifies how attributes are to be set in the output objects of the phase.

The following is a fragment of a grammar for verb groups in the Basic Phrase Recognizer:

```
VG --> VG2 Adv* V-en:1;
       head = (obj 1);
       active = T;
       aspect = perf;;

VG2 --> VG1 ''have'';;

VG2 --> V[have]:1 (Not);
        tense = (tense 1);;

VG1 --> Modal:1 (Not) Adv*;
        tense = (tense 1);;

Not --> ''not'';
        negative = T;;
```

This covers a phrase like "could not really have left". `V-en` and `Adv` refer to words that are past participles and adverbs, respectively. `V[have]` indicates some form of the verb "have". The use of indices like ":1" allows us to access the attributes of terminal symbols. The semantics in these rules sets the features of `active`, `aspect`, `tense`, and `negative` appropriately, and sets `head` to point to the input object providing the past participle.

The following is one rule in a grammar for the Clause-Level Event Recognizer for the labor negotiations domain used in the dry run of MUC-6 in April 1995.

```
Event -->
    Event-Adj*  NG[org]:1 (Compl)
    VG[active,resume-word]:2  NG[talk-word]
    {''with'' NG[org]:3 | Event-Adj}*;
        type = Talk;
        parties = (List (obj 1) (obj 3));
        talk-status = Bargaining;;
```

This says that when an organization resumes talks with an organization, it is a significant event. `Event-Adj` is matched by temporal, locative, epistemic and other adverbial adjuncts. `Compl` is matched by various possible noun complements. This rule creates an event structure in which the event type is `Talk`, the parties are the subject and the object of "with" matched by the patterns, and the talk status is `Bargaining`.

FastSpec has made it immensely easier for us to specify grammars, and recently it has become

one of the principal influences on the Tipster effort to develop a community-wide Common Pattern-Specification Language.

4 Compile-Time Transformations

For an application in which we had to recognize the products made by companies, we would want a pattern that would recognize

GM manufactures cars.

But in addition to writing a rule for this pattern, we would have to write rules for all the syntactic variations of the simple active clause, to recognize

Cars are manufactured by GM.
... GM, which manufactures cars.
... cars, which are manufactured by GM.
... cars manufactured by GM.
GM is to manufacture cars.
Cars are to be manufactured by GM.
GM is a car manufacturer.

Moreover, in each of these patterns we would need to allow the occurrence of temporal, locative, and other adverbials. Yet all of these variations are predictable, and every time we want the first pattern we want the others as well.

This consideration led us to implement what can be called "compile-time transformations". Expensive operations of transformation are not done while the text is being processed. Instead, the transformed patterns are generated when the grammar is compiled. We have implemented a number of parameterized metarules that specify the possible linguistic variations of the simple active clause, expressed in terms of the subject, verb, and object of the active clause, and having the same semantics. Then domain-specific patterns are defined that provide particular instantiations of the metarules.

The metarule for the basic active clause, as in "The company resumed talks", is

```
Event -->
    Event-Adj* NG[??subj]:1
    VG[active,??head]:2 NG[??obj]:3
    {P[??prep] NG[??pobj]:4 | Event-Adj}*;
    ??semantics;;
```

Once the variables ??subj, ??head, ??obj, ??prep, and ??pobj are defined by the user, they are plugged into this rule and a new specific rule is generated. Each of these variables is a (list of) lexical or other attributes, and when they are plugged into the metarule, they define a pattern that is constrained to those attributes. Adverbials are recognized by matching a sequence of input objects with Event-Adj. Indices are associated with each of the arguments of the head's predication, and these can be used in the semantics specified for particular pattern.

The metarule for passives, as in "Talks were resumed", is

```
Event -->
    NG[??obj]:3 VG[passive,??head]:2
    {P[??prep] NG[??pobj]:4 | Event-Adj}*;
    ??semantics;;
```

The object still has the index 3, so that the same semantics can be used for the passive as for the active.

The metarule for relative clauses with a gapped subject, as in "the company, which resumed talks ...", is

```
Event -->
    NG[??subj]:1 P[relpro]
    VG[active,??head]:2 NG[??obj]:3
    {P[??prep] NG[??pobj]:4 | Event-Adj}*;
    ??semantics;;
```

The metarule for nominalizations, as in "the company's resumption of talks", must appear in the Complex Phrase Recognizer and has the form

```
ComplexNG -->
    (NG[??subj]:1 P[gen]) NG[??head]:2
    (''of'' NG[??obj]:3)
    {P[??prep] NG[??pobj]:4 | Event-Adj}*;
    ??semantics;;
```

Here all the arguments are optional. We could simply have the bare nominal.

In addition to the basic patterns, middle verbs and symmetric verbs are handled. Middle verbs are verbs whose object can appear in the subject position and still have an active verb.

They resumed the talks.
The talks resumed.

The metarule that implements the middle "transformation" is as follows:

```
Event -->
    NG[??obj]:3 VG[active,??head]:2
    {P[??prep] NG[??pobj]:4 | Event-Adj}*;
    ??semantics;;
```

Symmetric verbs are verbs where an argument linked to the head with the preposition "with" can be moved into a subject position, conjoined with the subject. For example,

> The union met with the company.
> The union and the company met.
> The meeting between the union and the company.

To handle this there are patterns in the Complex Phrase Recognizer that recognize a conjunction of the subject and the prepositional argument, when the verb is designated symmetrical:

```
NG[??subj] ''and'' NG[??pobj]
```

This is then given a special attribute `symconj`, and in the Clause-Level Event Recognition phase, complex noun groups with this property are sought as subjects for symmetric verbs.

```
Event -->
    Event-Adj* NG[symconj]
    VG[active,??head]:2 NG[??obj]:3
    Event-Adj*;
    ??semantics;;
```

With this set of metarules, defining the necessary patterns becomes very easy. One need only specify the subject, verb, object, preposition, and prepositional object, and the classes of metarules that need to be instantiated, and the specific rules are automatically generated. For example, the specification for "resume" would be

```
Transformations:  Middle, Basic:
    1: Subj = org;
    2: Head = resume-word;
    3: Obj = talk-word;
       Prep = ''with'';
    4: PObj = org;
Semantics =
  <type = Talk;
   parties = (list (obj 1) (obj 4));
   talk-status = Bargaining;;>;
```

In the semantics, we set the type of event to be `Talk` and the talk status to be `Bargaining`. The parties are those referred to by the subject (1) and the prepositional object (4).

Our experience with this aspect of the FASTUS system has been very encouraging. During the preparation for MUC-6, it took us only about one day to implement the necessary clause-level domain patterns, because of the compile-time transformations.

5 Atomic versus Molecular Approaches

There are two approaches that have emerged in our experience with FASTUS. They might be called the "atomic" approach and the "molecular" approach. Both approaches are made easier by FastSpec and the compile-time transformations.

In the atomic approach, the system recognizes entities of a certain highly restricted type and assumes that they play a particular role in a particular event, based on that type; then after event merging it is determined whether enough information as been accumulated for this to be an event of interest. This approach is more noun-driven, and its patterns are much looser. It is most appropriate when the entity type is highly predictive of its role in the event. The microelectronics domain of MUC-5 and the labor negotiations were of this character. When one sees a union, it can only go into the union slot of a negotiation event.

In the molecular approach, the system must recognize a description of the entire event, not just the participants in the event. This approach is more verb-driven, and the patterns tend to be tighter. It is most appropriate when the syntactic role of an NP is the primary determinate of the entity's role in the event. The terrorist domain of MUC-3 and MUC-4, the joint venture domain of MUC-5 and the management succession domain of MUC-6 were of this character. You can't tell from the fact that an entity is a person whether he is going into or out of a position at an organization. You have to see how that person relates to which verb.

The distinction between these two approaches can be used as a conceptual tool for analyzing new

domains.

6 Adapting Rules from Examples

The FastSpec language and the compile-time transformations make it easier for linguists and computer scientists to define patterns. But they do not enable ordinary users to specify their own patterns. One way to achieve this would be to have automatic learning of patterns from examples provided by the user. We have begun in a small way to implement such an approach.

We need a way for the user to supply a mapping from strings in the text to entries in the template. This can be accomplished by having a two-window editor; the text being annotated or analyzed is in one window, the template in the other. The user marks a string in the text, and then either copies the string to a template entry or enters the set fill that is triggered by the string. Such a system is first of all a convenient text editor for filling data bases from text by hand. But if the system is trying to deduce the implicit rules the user is responding to to make the fills, then the system is automatically constructing an information extraction system as well.

We have implemented a preliminary experimental version of such a system, and are currently developing a more advanced one. We assume that the user somehow provides a mapping from text strings to template entries and that the semantics of the rule is completely specified by such a mapping. Moreover, we are only handling the case where the new rule to be induced is a specialization of an already existing rule, in the sense that

<Location> "-" "based"

is a specialization of

<Noun> "-" <Past-Participle>

In general, the problem of rule induction is very hard. What we are doing is a tractable and useful special case.

The first problem is to identify the phase in which the new rule should be defined. To do this, we identify the highest-level phase (call it Phase n) in which the constituent boundaries produced by the phase correspond to the way the user has broken up the text. A new rule is then hypothesized in Phase $n+1$. For example, if the user has marked the string "the union resumed talks with the company" and placed "the union" in one slot and "the company" in another, then Phase n is the Complex Phrase Recognizer, since it provides those noun groups as independent objects. On the other hand, if the string is "the union's resumption of talks with the company", then the Complex Phrase Recognizer will not do, since it combines at least "the union" and possibly "the company" into the same complex noun group as "resumption". We have to back up one more phase, to the Basic Phrase Recognizer, to get these noun groups as independent elements.

In the current version, we determine what Phase $n+1$ rule matches the entire string and then construct as general as possible a specialization of that rule. For the semantics of the specialized rule, we encode the mapping the user has constructed.

Determining the correct level of generalization of the hypothesized rule is a difficult problem. There are some obvious heuristics that we have implemented, such as generalizing "38" to Number and "Garrick" to Person. But should we generalize "United Steel Workers" to Union or to Organization? Our current approach is to be conservative and to experiment with various options.

Once the rule is hypothesized it will be presented to the user in some form for feedback and validation. How best to implement this is still a research issue.

This work represents a productive synergy between the Tipster project and another FASTUS-based project at SRI, the Message Handler, for processing a large number of types of military messages.[1] The basic ideas were worked out in connection with our Tipster II project. We will be developing a sophisticated, general version of the system as part of our Tipster III research. In the meantime, we are using the theory that we have worked out to develop a restricted learning component for the Message Handler. This effort of applying theory to a very complex real-world task can give us insights into the various problems that arise.

7 Coreference Resolution

There are three places in FASTUS processing that coreference resolution gets done. Early in the processing, in Name Recognition, entities that are referred to by the same name, or by a name and a plausible acronym or alias, are marked as coreferential. Late in the processing, in Event Merging,

[1]This project is being carried out in collaboration with E-Systems, Greenville, and is funded by the Defense Advanced Research Project Agency through the US Army Topographic Engineering Center under contract no. DACA76-93-L-0019.

some coreference resolution happens as a side-effect of merging event strutures. In the example of Section 2, we learn from Clause-Level Event Recognition that Garrick will become president and COO, and we learn that "he" will succeed Costello. These are two consistent management succession event descriptions, so they are merged, and in the course of doing so, we resolve "he" to Garrick.

The third type of coreference resolution occurs after complex noun groups are recognized. This module was implemented early in 1995 in order to participate in the Coreference evaluation in MUC-6, but it was done in a way that was completely in accord with normal FASTUS processing, and the results of coreference resolution are used by subsequent phases.

Coreference resolution is done only for definite noun groups and pronouns. We experimented with an algorithm for bare noun groups, but it hurt precision more than it helped recall.

Two principal techniques are used to resolve definite noun groups. First we look for a previous noun group with the same head noun. thus, "the agreement" will resolve with "an agreement". In addition, we look for a previous object of the right domain-specific type. Thus, "the Detroit automaker" will resolve to "General Motors" or to "a company", since "automaker" is of type COMPANY and General Motors is a company. No use is made of synonymy or of a sort hierarchy otherwise. Thus, "the agreement" will not resolve back to "a contract". This is obviously a place where the algorithm can be improved. Rather arbitrarily, we have set the search window to ten sentences; this is a parameter than can be experimented with.

For third person pronouns we use an approximation of the algorithms of Hobbs (1978) and Kameyama (1986). We search for noun groups of the right number and gender, first from left to right in the current sentence, then from left to right in the previous sentence, and then from right to left in two more sentences. The pronoun "they" can be identified with either a plural noun group or an organization.

For singular first person pronouns, "I" and "me", we resolve to the nearest person. For plural first person pronouns, "we" and "us", we resolve to the nearest organization or set of persons. We allow all of the current sentence, including material to the right of the pronoun, since quotes frequently precede the designation of the speaker, as in

"I was robbed," said John.

An obvious improvement would be to determine whether the person occurs as the subject of a verb of speaking, but an informal examination of the data suggested this would not result in a significant improvement.

The heuristics we use for coreference resolution are very simple and easily implemented in a FASTUS framework. Numerous improvements readily suggest themselves. But we have been surprised how strong a performance can be achieved just with these simple heuristics. Our performance on the MUC-6 Coreference task was a recall of 59% and a precision of 72%. These scores placed SRI among the leaders.

8 Information Extraction and Document Retrieval

As part of Tipster II.V, we are engaged in a joint effort with the University of Massachusetts to determine ways in which information extraction technology can improve the performance of document retrieval systems, such as the INQUERY system. Initially, we are pursuing three investigations.

1. The first is simply to examine a large number of highly ranked false positives for a number of queries, and to determine whether information extraction techniques can help. We have done this on a small scale, five texts for one TREC topic. The topic was actual retaliation against terrorists. The false positives all talked about retaliation against terrorists, but it was embedded in negative or modal contexts, such as the following:

... will not retaliate against the terrorist attack ...
... discussed the possibility of retaliating.
... if we retaliate against terrorists ...

These are the kinds of features that Basic and Complex Phrase Recognition in FASTUS can spot, and the texts could thereby be rejected.

2. We have already developed an information extraction system for the management succession domain, and that corresponds to one of the TREC topics. We will run INQUERY on that topic and then run the MUC-6 FASTUS system on the 100 texts that INQUERY ranks most highly. We can then determine whether there is any criterion definable in terms of the events extracted that can improve on INQUERY's ranking. This will lead to the question of how much information extraction domain development is necessary for how much corresponding document retrieval improvement.

3. We have a moderately well developed module for coreference resolution. Can this be used to improve INQUERY's performance? The idea is to apply FASTUS processing, up through coreference resolution, to all the documents in the corpus. We would then use the resulting coreference chains to increase the richness of concepts in the text. For example, consider two documents that each mention IBM once. The first is about IBM and contains numerous subsequent references to "the computer company" and "they". The second mentions IBM only in the context of IBM-compatible peripherals and is concerned with something else entirely. Having every reference to IBM count as a mention of IBM will result in the first document having a much higher score than the second. This method could help in both directions. If the topic concerns IBM, references to the computer company will increase the score. If the topic concerns computer companies, references to IBM will increase the score.

9 Conclusion

Under the auspices of the Tipster II program, we have developed in FASTUS a mature, effective, efficient, robust information extraction system. We have made it substantially easier to use in new domains by implementing the FastSpec declarative specification language and the compile-time transformations, and we believe our work on adapting rules from examples will make the system yet easier to use in new domains.

We have integrated the system into the developing uniform, modular Tipster architecture.

In our view the principal current problems are the need to handle broader domains and applications, the need to continue to make new domains easier to implement, and the need to use the technology in a wide variety of new applications.

Acknowledgments

The research described here was funded by the Defense Advanced Research Projects Agency under Office of Research and Development contract 94-F157700-000.

References

[1] Appelt, Douglas E., Jerry R. Hobbs, John Bear, David Israel, Megumi Kameyama, and Mabry Tyson, 1993. "The SRI MUC-5 JV-FASTUS Information Extraction System", *Proceedings*, Fifth Message Understanding Conference (MUC-5), Baltimore, Maryland, August 1993.

[2] Hobbs, Jerry R., 1978, "Resolving Pronoun References", *Lingua*, Vol. 44, pp. 311-338. Also in *Readings in Natural Language Processing*, B. Grosz, K. Sparck-Jones, and B. Webber, editors, pp. 339-352, Morgan Kaufmann Publishers, Los Altos, California.

[3] Hobbs, Jerry R., Douglas E. Appelt, John Bear, David Israel, and Mabry Tyson, 1992. "FASTUS: A System for Extracting Information from Natural-Language Text", SRI Technical Note 519, SRI International, Menlo Park, California, November 1992.

[4] Kameyama, Megumi, 1986. "A Property-sharing Constraint in Centering", in *Proceedings*, 24th Annual Meeting of the Association for Computational Linguistics, New York, NY, pp. 200-206.