Towards Automatic Animated Storyboarding

Patrick Ye♠ and Timothy Baldwin♠♡

♠ Computer Science and Software Engineering
University of Melbourne, Australia

♡ NICTA Victoria Laboratories
University of Melbourne, Australia

{jingy,tim}@csse.unimelb.edu.au

Abstract

In this paper, we propose a machine learning-based NLP system for automatically creating animated storyboards using the action descriptions of movie scripts. We focus particularly on the importance of verb semantics when generating graphics commands, and find that semantic role labelling boosts performance and is relatively robust to the effects of unseen verbs.

Introduction

Animated storyboards are computer animations created based on movie scripts, and used as crude previews by directors and actors during the pre-production of movies. Creating non-trivial animated storyboards is a time- and labour-intensive process, and requires a level of technical expertise that most people do not have. In this research, we propose to automate the process of animated storyboarding using a variety of NLP technologies, potentially saving time and money and also providing a dynamic, visual environment for script creation and fine-tuning.

The creation of an animated storyboard can be described as a two-step process. The first step is to construct a static virtual stage with virtual actors and props to approximate the scene to be shot. The second step is to create the interactions between the virtual actors and props, to visualize the events depicted by the action descriptions of the movie scripts. This research is focused on the second step as it is more labour-intensive and technically challenging than the first step.

There are three major differences between existing NLP-aided animation systems and our system. Firstly, most existing systems use handcrafted rules to map the results of language analysis onto graphics commands, whereas our system uses a machine learning system to perform this task automatically. Secondly, existing systems were designed for domain-specific tasks with a controlled vocabulary and syntax, whereas our system is open-domain with no restrictions on the language used other than that the input text is in the style of 3rd person narration. Thirdly, existing systems are all coupled with customised graphics systems designed specifically for their respective tasks, whereas our system is designed to interface with any graphics system that offers access through a programming language style interface.[1]

[1]We used a free animation system called Blender3D (`www.blender.org`) in our experiments.

Since the main purpose of animated storyboards is to visualise the actions described in the scripts, and the actions are mostly expressed in verb phrases, the linguistic analysis in our system is focused on verb semantics. Several NLP techniques are used to perform this analysis, namely POS tagging is used to identify the verbs, and semantic role labelling (SRL) (Gildea and Jurafsky 2002) is used to identify the semantic roles of each verb. Furthermore, our linguistic analysis also makes use of lexical resources such as WordNet (Fellbaum 1998).

The main contribution of this research is the proposal of a novel multimodal domain of application for NLP, and the integration of NLP techniques with extra-linguistic and multimodal information defined by the virtual stage. We are especially interested in the usefulness of SRL in our task since it is an intuitively useful NLP technique but has rarely found any practical applications. We are also interested in how well our system could handle verbs not seen in the training data, since rule-based systems often cannot handle such verbs.

Related Work

Most existing literature on applying NLP techniques to computer animation is limited to domain-specific systems where a customised computer graphics module is driven by a simplistic handcrafted NLP module which extracts features from input text. The pioneering system to support natural language commands to a virtual world was SHRDLU (Winograd 1972), which uses handcrafted rules to interpret natural language commands for manipulating simple geometric objects in a customised block world. The K3 project (Funakoshi, Tokunaga, and Tanaka 2006) uses handcrafted rules to interpret Japanese voice commands to control virtual avatars in a customised 3D environment. Carsim (Johansson et al. 2004; 2005) uses a customised semantic role labeller to extract car crash information from Swedish newspaper text and visually recreate the crash using a customised graphics system.

While existing systems have been successful in their own respective domains, this general approach is inappropriate for the task of animated storyboarding because the input language and the domain (i.e. all movie genres) are too broad. Instead, we: (a) use a general-purpose graphics system with a general set of graphics commands not tuned to any one domain, and (b) we use automated NLP and machine learning techniques to perform language analysis on the input scripts, and control the graphics system.

Ye, Patrick and Timothy Baldwin (2008) Towards Automatic Animated Storyboarding, In Proceedings of the 23rd Conference on Artificial Intelligence (AAAI-08), Chicago, USA, pp. 578-583.

Our work is also similar to machine translation (MT) in the sense that we aim to translate a source language into a target language. The major difference is that for MT, the target language is a human language, and for our work, the target language is a programming language. Programming languages are much more structured than natural languages, meaning that there is zero tolerance for disfluencies in the target language, as they result in the graphics system being unable to render the scene. Also, the lack of any means of underspecification or ambiguity in the graphics commands means that additional analysis must be performed on the source language to fully disambiguate the language.

Overview

In this section, we describe our movie script corpus, then provide detailed discussion of the virtual stages and how we use them in our experiments, and finally outline the main computational challenges in automatic animated storyboarding.

Movie Script Corpus

Movie scripts generally contain three types of text: the scene description, the dialogue, and the action description. In this research, the only section of the script used for language analysis is the acting instructions of the action descriptions, as in (1).[2] That is, we only visualise the physical movements of the virtual actors/props.

(1) Andy's hand lowers a ceramic piggy bank in front of Mr. Potato Head and shakes out a pile of coins to the floor. Mr. Potato Head kisses the coins.

In real movies, dialogues are of course accompanied by body and facial movements. Such movements are almost never specified explicitly in the movie scripts, and require artistic interpretation on the part of the actual actors. They are therefore outside the scope of this research.

We animated around 95% of the Toy Story script using 21 base graphics commands. The script was split up according to its original scene structure. For each annotated scene, we first constructed a virtual stage that resembles the corresponding scene in the original movie. We then performed semantic role labelling of that scene's acting instructions, determined the correct order for visualisation, annotated every visualisable verb with a set of graphics commands and finally, annotated the correct grounding of each argument. In total, 1264 instances of 307 verbs were annotated, with an average of 2.84 base graphics commands per verb token.

Virtual Stage

A novel and important aspect of this research is that we use the physical properties and constraints of the virtual stage to improve the language analysis of the acting instructions. Below, we outline what sort of information the virtual stage provides.

The building blocks of our virtual stages are individual 3D graphical models of real world objects. Each 3D model is hand-assigned a WordNet synset in a database. The virtual stages are assembled through a drag-and-drop interface between the database and the graphics system.

Each 3D model in the virtual stage can be annotated with additional WordNet synsets, which is useful for finding entities that have multiple roles in a movie. For example, the character Woody in Toy Story is both a *toy*, i.e. "an artifact designed to be played with", and a person. Furthermore, each 3D model in the virtual stage can be given one or more name, which is useful for finding entities with more than one title. For example, in the script of Toy Story, the character of Mrs. Davis is often referred to as either *Mrs. Davis* or *Andy's mum*.

WordNet 2.1 synsets were used to annotate the 3D models. Furthermore, since WordNet provides meronyms for its noun synsets, it will be possible to further annotate the components of each 3D model with the corresponding WordNet synsets, facilitating the incorporation of more real world knowledge into our system. For example, consider the sentence *Andy sits in the chair*. Since a chair (typically) consists of a seat, legs, and a back, it would be useful if the system could differentiate the individual components of the 3D model and avoid unnatural visualisation of the sentence, such as Andy sitting on the back of the chair instead of the seat.

Finally, an important feature of the virtual stages is that the status (position, orientation, etc) of any virtual object can be queried at any time. This feature makes it possible to extract extra-linguistic information about the virtual objects during the animation process.

Main Computational Tasks

As stated above, this research centres around extracting verb semantic information from the acting instructions and the virtual stage, and then using this information to create an appropriate animation.

We identified two main computational tasks for our research: extracting the timing relationships between the verbs in the input script, and constructing graphics commands using the verb semantic and virtual stage information.

Since a correct storyboard requires the actions described by the movie scripts to be animated in the correct chronological order, the extraction of the timing relationships between them is the first task of our system. Generally, verbs should be visualised in the order they appear in the text. For instance, in (1), *lower* should be visualised first, followed by *shake*, and finally *kiss*. However, there are cases were the order of appearance of the verbs does not correspond to the order of visualisation, as in (2) where *pummel* and *bark* should be visualised at the same time rather than sequentially in the order they appear.

(2) Sid pummels the figure with rocks while Scud is barking wildly at it.

The second task of constructing graphics commands is considerably more complicated than the first task, and is the main subject of this paper. This task consists of 2 subtasks: selecting the graphics instructions for the target verbs, and constructing the arguments for the cho-

[2]All the movie script examples in this paper are taken from a modified version of the Toy Story script.

sen graphics instructions. The graphics instructions used in this research are similar to procedures in procedural programming languages, and have the general form $command(arg0, arg1, ..., argN)$. Therefore, the first subtask is to decide the value of $command$, and the second subtask is to decide on the values of each argument to the chosen command.

The linguistic analysis involved in the task of constructing graphics commands is mostly focused on the extraction of verb semantics. The main NLP technique used here is semantic role labelling. However, as semantic role labellers only identify the surface string associated with each verb argument, an additional step is required to ground the surface string associated with each argument to one or more specific virtual stage object. This grounding task is relatively less studied, and is far from a trivial problem. For instance, consider *The toys on the floor* in (3). This does not simply correspond to all virtual objects that are annotated with the synset *toy*[1] in the virtual stage, but has the extra constraints of the objects needing to be on the floor and salient to the current area of focus.

(3) The toys on the floor all stop and run towards the monitor.

Classifier Setup

In this section, we provide details of how we apply machine learning to the task of choosing the graphics commands.

Since we use a generic graphics system, it is uncommon for a verb to be fully visualised with a single base graphics command, and very often, it is necessary to use a sequence of combinations of base graphics commands to visualise a verb. Overall, the average number of base graphics commands used to visualise a single verb was 2.84.

In the context of the classifier, we will refer to these combinations of base graphics commands as "actions". Base graphics commands within a single action are executed from the same time stamp, sequentially at the beginning of each animation cycle.

We treat the subtask of assigning sequences of actions to verbs as a Markov process, and the overall task of action generation as a classification problem. The feature vector of the classifier denotes the latest semantic and virtual stage information, and the class labels denote the next action to be performed. For example, suppose the sentence to be visualised is *The man grabs the mug on the table*, and the virtual world is configured as in Figure 1a. The classifier should select a set of base commands (i.e. an action) which make the virtual man stand up (Figure 1b), move close to the virtual mug (Figure 1c), and extend his hand to the mug (Figure 1d). Finally, the classifier should select an `end-of-sequence` action signalling the completion of the visualisation. Each of these movements needs to be executed and the virtual stage updated, and this new information should be used to condition the selection/animation of the next movement.

In the remainder of this section, we present the features that we used in our experiments.

Linguistic Features

The linguistic features used in our system can be divided into the following categories:

Verb Features: These features include the original form of the target verb, its lemmas and WordNet synsets, and the hypernyms of these synsets. The inclusion of hypernyms is intended to provide the classifiers with the means to generalise when dealing with previously unseen verbs. The verbs are not disambiguated; instead all WordNet senses of the target verb are included.

Collocation Features: These are the lemmas of all the open class words (determined according to the POS tagger output) that occur in the same sentence as the target verb.

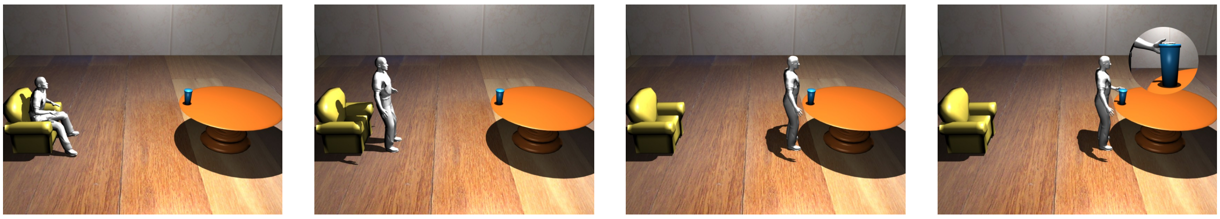
Semantic Role Features: These features include: the types of roles that the target verb takes (`ARG0`, `ARGM-LOC`, etc.); the WordNet synsets of the head words of the grounded constituents combined with the semantic roles of the constituents; and a series of syntactic-collocational features of the constituents of the target verb. Note that we have hand-resolved all anaphora in the data (around 35% of the sentences in the training data contained one or more anaphora), and used this oracle-style anaphora resolution for all methods presented in this paper.

The syntactic-collocational features of the constituents were designed to capture the details missed by the grounded semantic roles. The purpose of grounding semantic roles is to find the most relevant virtual object with respect to given textual forms of constituents, which is not sufficient if the most relevant virtual object does not exactly correspond to the semantics of the constituent. For example, consider `ARG0` and `ARGM_LOC` of the SRLed sentence in (4). The head of `ARG0` is *hand* which corresponds to a part of the body of *Andy*. However, since grounding is performed only at the level of virtual objects, `ARG0` is grounded to the virtual *Andy*, and the head of the constituent is lost in the grounding process. Similarly for `ARGM_LOC`, the constituent is a PP, and the most relevant virtual object is the virtual *Mr. Potato Head*. This is hence what this phrase will be grounded to, resulting in the additional locational information of *in front of* being lost. The syntactic-collocational features reinstate the linguistic content of each constituent to the feature space, providing the classifier with the means to fine-tune the grounding information and more faithfully render the virtual scene.

(4) [$_{\texttt{ARG0}}$ Andy's hand] [$_{\texttt{TARGET}}$ lowers] [$_{\texttt{ARG1}}$ a ceramic piggy bank] [$_{\texttt{ARG\_LOC}}$ in front of Mr. Potato Head]

The syntactic-collocational features are collected recursively from the root of the smallest parse subtree covering all the tokens in the corresponding constituent. Only semantic roles which are prepositional phrases (PP), noun phrases (NP), adjective phrases (ADJP) and adverb phrases (ADVP) are used to generate these features.

If the semantic role is a PP, it will generate a feature that includes the head preposition and the recursively retrieved features of its argument. If the constituent is an NP, it will generate a feature that includes the WordNet synset of the



1a 1b 1c 1d

Figure 1: Examples of the sequence of virtual stages corresponding to *The man grabs the mug on the table*

head noun and the recursively retrieved features of its possessor and modifiers. If the semantic role is an ADJP or ADVP, it will generate a feature that includes its head, and all the recursively retrieved features of its modifiers. Take the constituent *in front of Mr. Potato Head*, for example. *Mr. Potato Head* would generate the feature of (NP, toy^1), and *front of Mr. Potato Head* would generate the feature of (NP, front^2, (PP_MOD, PP, *of*, (ARG, (NP, toy^1)))).

The ASSERT system (Pradhan et al. 2004) was used to perform the semantic role labelling, the Charniak reranking parser (Charniak and Johnson 2005) was used to perform the parsing, and the SVMTool (Gimnez and Mrquez 2004) POS Tagger (version 1.2.2) was used to POS-tag the input text.

Virtual Stage Features

The virtual stage features are binary features designed to capture the spatial relationships between the grounded constituents. The general spatial relationships are denoted as the results of different proximity and orientation tests between the virtual objects. For virtual objects that contain bones, the spatial relationships also include the results of the proximity tests between their bones and other virtual objects.

In addition, the virtual stage features include description of each ongoing base graphics command that each grounded virtual object is involved in.

Experiments

The main questions we address in our experiments are: (1) What is the effect of unseen verbs on overall performance? (2) Does semantic role labelling positively contribute to classification accuracy? and (3) How reliant is the classification on accurate grounding?

In all the experiments, we divided our original datasets into training, development and test sets. The training set contains roughly the same number of examples as the combined development and test sets, and the development and test sets are of roughly the same size. We used a maximum entropy (Ratnaparkhi 1996) classifier to generate models from the training data.[3] The parameters of these models are then adjusted using the development data, and finally tested on the test data. The details of the parameter tuning can be found in (Ye and Baldwin 2006).

To explore the first question of the impact of unseen verbs on overall performance, we performed 2 sets of experiments.

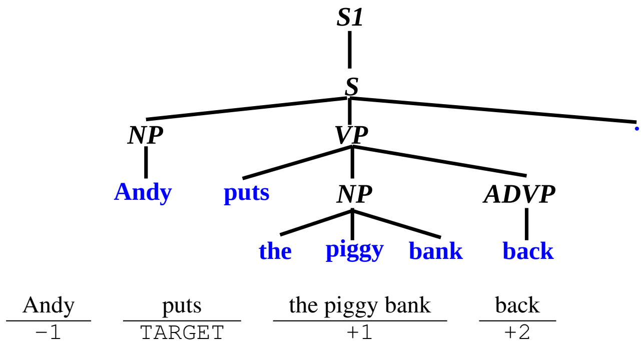


Figure 2: Example of substitute semantic roles

The first set of experiments was designed to test how well the classifiers work when the verbs in the test data are included in the training data whenever possible, i.e. when verbs instances are evenly distributed. The second set of experiments was designed to test how well the classifiers work when all the verbs in the test data are randomly chosen to not appear in the training data.

To explore the second question of the impact of SRL, we experimented with three different SRL methods: using gold standard semantic roles, using an automatic labeller (ASSERT), and using simple parse features as a replacement for semantic roles.

The gold standard semantic role data was created by hand-correcting the outputs of ASSERT, and the automatically labelled semantic roles were then evaluated using the evaluation script for the CoNLL 2004 SRL shared task (Carreras and Màrquez 2004). Overall, ASSERT had a precision of .744, a recall of .564 and an F-score of .642. For comparison, over the CoNLL 2004 data, ASSERT had a precision of .724, a recall of .668 and an F-score of .695. It can be observed that the recall over the movie script corpus is significantly lower than the data it was trained for. We also observed that many of the missing semantic roles were caused by the POS tagger, which mis-classified several high frequency verbs (e.g. *grab*, *jump* and *climb*) as nouns.

When no semantic roles are used, we used a parse tree-based method to extract top-level NPs, PPs, ADJPs and ADVPs from the output of the Charniak parser, and used the relative positions of these phrases with respect to the target verb as substitute semantic roles. Figure 2 shows the substitute semantic roles for the sentence *Andy puts the piggy*

[3]The learner is downloadable from: `homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html`

SRL	Precision	Recall	F-score
Gold	.508	.701	.589
Auto	.470	.490	.480

Table 1: Grounding results

bank back.

The final question of grounding performance was explored via two sources of grounding information: (1) gold standard grounding annotations based on the gold standard semantic roles; and (2) automatic grounding predictions based on either the gold standard semantic roles, the outputs of ASSERT or the parse tree-based phrase extractor.

The automatic grounding predictions were obtained using a string matching method. Given the text descriptor of an object in the text, this method first tries to match it against the user-specified names of the virtual objects, and if no virtual objects match, it then matches the string against the WordNet synsets of the virtual objects. For example, if the virtual stage contains a virtual actor named Woody, and the string *Woody* or *Woody's hand* is to be matched, the automatic grounding module will correctly return the virtual Woody. On the other hand, if the virtual stage contains multiple virtual toys, and the string *toy* is to be matched, the automatic grounding module would incorrectly return all the virtual toys. Table 1 shows the results of the automatic grounding method on the gold standard semantic roles and automatically-labelled semantic roles.

We evaluate our models in terms of classification accuracy with respect to the gold-standard action annotations for each verb, on a per-action basis. The majority class baseline for our classifiers is the overall most frequent action (i.e. `end-of-sequence`).

Results and Analysis

Experiment Set 1: Balanced Distribution

In this first set of experiments, we partitioned the training, development and test datasets such that the annotated verb instances were stratified across the three datasets. The stratification ensures the distributions of verbs in the training, development and test sets are as similar as possible. Our "Classifier" performance is based on a classifier which has been trained over all the verbs in the training set. Table 2 shows the classification accuracies under different combinations of gold-standard and automatic grounding, and gold-standard, automatic and parser-emulated SRL. The baseline of these classifiers is .386.

All our classifiers outperformed the majority class baseline. Unsurprisingly, classifiers trained on both the gold-standard SRL and the gold-standard grounding data achieved the highest accuracy, although the relative difference in including gold-standard SRL was considerably less than was the case with the grounding.

	Source of Grounding	Source of SRL		
		Gold	Auto	Parser
Classifier	Gold	.568	N/A	N/A
	Auto	.531	.526	.475

Table 2: Accuracy for experiment set 1 (no unseen verbs)

	Source of Grounding	Source of SRL		
		Gold	Auto	Parser
Classifier	Gold	.536	N/A	N/A
	Auto	.479	.452	.463

Table 3: Accuracy for experiment set 2 (all unseen verbs)

These experiments show that SRL can indeed be useful in the domain of automatic animated storyboarding.

It is perhaps slightly surprising that the classifier based on automatically grounded gold-standard semantic roles only slightly outperformed the one based on automatic SRL. A closer look at the results showed that the gold-standard grounding annotation only identified 1266 grounded semantic roles, but the automatic grounding method came up with 1748 grounded semantic roles. This overgeneration of grounded semantic roles introduced a massive amount of noise into the data, thereby lowering the performance of the corresponding classifier.

On the other hand, it was observed that ASSERT tended to make the same mistakes for the same verbs. Hence, the same SRL mistakes were consistently present in both the training data and the test data, thereby not causing significant negative effects.

These experiments show that there is significant room for improvement in the grounding of the semantic roles. The biggest improvement among the SRL based classifiers came when gold-standard grounding annotation was used. This indicates that the string-matching grounding system we are currently using is inadequate, and deeper linguistic analysis is needed.

Experiment Set 2: All Unseen Verbs

Recall from above that all the hypernyms of all the senses of the target verbs were included in the verb features, in the hope that they could provide some generalisation power for unseen verbs. The purpose of this second set of experiments is to test how well our classifiers can generalise verb semantics.

We first randomly divided our datasets into 4 portions, each containing roughly the same number of verbs, with all instances of a given verb occurring in a single portion. We then performed 4-fold cross-validation, using two portions as training, one as development and one as test data. In our results, we report the average classification accuracy.

Table 3 shows the classification accuracies under the same combinations of SRL and grounding data as for experiment set 1. The majority class baseline for these classifiers is .393

All the classifiers in this set of experiments performed worse than in the first set of experiments. This is not surprising given that we do not have direct access to the semantics of a given verb, as it never occurs directly in the training data. It is also encouraging to see that despite the performance drop, all the classifiers still outperformed the baseline.

The classifiers based on automatically-grounded semantic roles suffered the greatest drop in performance: the automatic SRL-based classifier performed below the parser-emulated SRL based classifier, even. This is not surprising because most of the verb semantic information used by these classifiers is provided by the grounded semantic roles. In the first set of experiments, even though the semantic role labeller didn't perform well, at least the errors were consistent between the training and test sets, and this consistency to some degree offset the poor semantic role labelling. However, in this set of experiments, the errors in automatically obtained semantic roles are no longer consistent in the training and test sets, and only the correctly-labelled semantic roles generalise well to unseen verbs. This is why the classifier based on the gold-standard grounding annotations suffered the least performance drop among all the semantic role-based classifiers, whereas the classifier based on automatic SRL suffered the most.

On the other hand, the parser-emulated semantic roles are not greatly affected by the unseen verbs, because these semantic roles only depend on the relative positions of the relevant phrases to the target verb. These relative positions tend to be much more verb-independent than the real semantic roles, and are therefore less affected by the variation of the target verbs.

Discussion and Conclusion

Three observations can be made from the experiments. Firstly, unseen verbs have a noticeable negative impact on the overall performance of the classifier, especially when the semantic roles are not of high standard. However, unseen verbs did not cause the classifiers to completely fail, which is encouraging as it shows that our method is relatively robust over unseen verbs, unlike rule-based systems which rely on explicit information for each verb.

Secondly, semantic role labelling contributes positively to our task. However, it needs to achieve higher performance in order to be consistently useful when unseen verbs are involved.

Thirdly, the performance of the classification relies heavily on the grounding of the semantic roles, and the string matching-based grounding method tends to overgenerate groundings, which in turn introduce noise into our data and reduce the effectiveness of the resultant classifier.

In conclusion, we have presented a novel machine learning method for automatically generating animated storyboards. Our method uses several NLP techniques and resources, including parsing, SRL, and WordNet. The experiments presented in this paper show that the features we used are effective and can generalise over previously unseen verbs. Furthermore, we believe that the inclusion of a virtual stage (i.e. encoding of real world interaction) provides a novel perspective to the application and evaluation of NLP techniques, as demonstrated by the use of SRL and parsing in our experiments.

Our short term future work will be focused on the grounding process of the semantic roles. We will build a more complete grounding module which is capable of resolving complex NPs and PPs with the aid of the virtual stage, and we will investigate how current techniques in word sense disambiguation, anaphora resolution and co-reference resolution can be incorporated with the grounding process to provide a more integrated solution.

References

Carreras, X., and Màrquez, L. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proc. of the 8th Conference on Natural Language Learning (CoNLL-2004)*, 89–97.

Charniak, E., and Johnson, M. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, 173–180.

Fellbaum, C., ed. 1998. *WordNet: An Electronic Lexical Database*. Cambridge, USA: MIT Press.

Funakoshi, K.; Tokunaga, T.; and Tanaka, H. 2006. Conversational animated agent system k3. In *Proceedings of 2006 International Conference on Intelligent User Interfaces (IUI2006), Demo Session*.

Gildea, D., and Jurafsky, D. 2002. Automatic labeling of semantic roles. *Computational Linguistics* 28(3):245–288.

Gimnez, J., and Mrquez, L. 2004. Svmtool: A general pos tagger generator based on support vector machines. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, 43–46.

Johansson, R.; Williams, D.; Berglund, A.; and Nugues, P. 2004. Carsim: A System to Visualize Written Road Accident Reports as Animated 3D Scenes. In *ACL2004: Second Workshop on Text Meaning and Interpretation*, 57–64.

Johansson, R.; Berglund, A.; Danielsson, M.; and Nugues, P. 2005. Automatic text-to-scene conversion in the traffic accident domain. In *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, 1073–1078.

Pradhan, S.; Hacioglu, K.; Krugler, V.; Ward, W.; Martin, J. H.; and Jurafsky, D. 2004. Support vector learning for semantic argument classification. *Machine Learning* 60(1–3):11–39.

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

Winograd, T. 1972. *Understanding Natural Language*. Academic Press.

Ye, P., and Baldwin, T. 2006. Semantic role labeling of prepositional phrases. *ACM Transactions on Asian Language Information Processing (TALIP)* 5(3):228–244.