Toward Automatic Character Identification in Unannotated Narrative Text

Josep Valls-Vargas[1], Santiago Ontañón[1] and **Jichen Zhu[2]**
[1]Computer Science, [2]Digital Media
Drexel University
Philadelphia, PA 19104
josep.vallsvargas@drexel.edu, santi@cs.drexel.edu, jichen.zhu@drexel.edu

Abstract

We present a case-based approach to character identification in natural language text in the context of our *Voz* system. *Voz* first extracts entities from the text, and for each one of them, computes a feature-vector using both linguistic information and external knowledge. We propose a new similarity measure called *Continuous Jaccard* that exploits those feature-vectors to compute the similarity between a given entity and those in the case-base, and thus determine which entities are characters or not. We evaluate our approach by comparing it with different similarity measures and feature sets. Results show an identification accuracy of up to 93.49%, significantly higher than recent related work.

Introduction

Computational narrative systems, especially story generation systems, require the narrative world to be encoded in some form of structured knowledge representation formalism (Bringsjord and Ferrucci 1999; Ontañón and Zhu 2011). Currently this representation is mostly hand-authored for most computational narrative systems. This is a notoriously time-consuming task requiring expertise in both storytelling and knowledge engineering. This well-known "authorial bottleneck" problem can be alleviated by automatically extracting information, at both linguistic and narrative levels, from existing written narrative text.

Except for a few pieces of work such as (Elson 2012; Finlayson 2008), automatically extracting structure-level narrative information directly from natural language text has not received much attention. We would like to further connect the research areas of computational narrative and Natural Language Processing (NLP) in order to develop methods that can automatically extract structural-level narrative information (e.g., Proppian functions) directly from text.

In this paper, we present our approach to automatically identifying characters from unannotated stories in natural language (English). Characters play a crucial role in stories; they make events happen and push the plot forward. Depending on the genre, people along with anthropomorphized animals and objects can all act as characters in a story. Being able to identify which entities in the text are characters is a necessary step toward our long-term goal of extracting structure-level narrative information such as character roles.

We present a case-based approach for character identification in the context of our system *Voz*. After extracting entities from the text, *Voz* computes a set of 193 features using both linguistic information in the text and external knowledge. We propose a new similarity measure called *Continuous Jaccard* to compute similarity between a given entity and those in the case-base of our system, and thus determine whether the new entity is a character or not. We evaluate our approach by comparing it with different similarity measures and feature sets. Results show an identification accuracy of 93.49%, a significant increase from recent related work.

The rest of this paper is organized as follows. We first present related work. Then we discuss our approach for case-based character identification. After presenting our dataset and feature set, we discuss our empirical evaluation. Finally we conclude and discuss directions of future work.

Related Work

Character identification, related to named entity recognition and nominal actor detection, is a crucial step toward narrative structure extraction. Goyal et al.'s AESOP system (2010) explored how to extract characters and their affect states from textual narrative in order to produce plot units (Lehnert 1981) for a subset of Aesop fables. The system used both domain-specific assumptions (e.g., only two characters per fable) and external knowledge (word lists and hypernym relations in WordNet) in its character identification stage. More recently, Calix et al. (2013) proposed an approach for detecting characters (called "sentient actors" in their work) in spoken stories based on features in the transcribed textual content using ConceptNet and speech patterns (e.g., pitch). Their system detects characters through supervised learning techniques and uses this information for improving document retrieval. The work presented in this paper follows this line of work, but we propose a case-based approach with an extended set of features and a new similarity measure, obtaining significantly better results.

Also relevant for the work presented in this paper is that of more general narrative structure extraction. Chambers and Jurafsky (2008) proposed using unsupervised induction to learn what they called "narrative event chains" from raw newswire text. In order to learn Schankian script-like information about the narrative world, they use unsupervised learning to detect the event structures as well as the roles

of their participants (characters) without pre-defined frames, roles, or tagged corpora (2009). On some related work, Regneri et al (2011) worked on the specific task of identifying matching participants in given scripts in natural language text using semantic and structural similarities and using Integer Linear Programming (Wolsey 2000). In the work presented in this paper, we adapted some of the features used by these more general narrative extraction systems for the task of character identification.

Case-based Character Identification

Our character identification method uses Case-Based Reasoning (CBR) (Aamodt and Plaza 1994), a family of algorithms that reuse past solutions to solve new problems. The previously solved problems, called cases, are stored in a case-base. In our approach, each case is an entity, represented as a feature-vector, already annotated as either *character* or *non-character* by a human.

In this paper, we address the following problem: given an entity e, extracted from an unannotated story s in natural language, determine whether e is a character in story s. In this context, "entities" could be characters, props, or other objects (potentially referred to by pronouns), and are defined as a given node in the parse tree of a sentence (and its associated subtree).

In a nutshell, for each new target entity to be classified, a feature-vector is constructed to compare the target entity with the entities in the case-base. The most similar entity in the case-base is *retrieved* and used to predict whether the target entity is a character or not. The key steps of this process are: *entity extraction*, *feature-vector construction* and *entity classification*. Below we discuss these processes in the context of our *Voz* system, which aims at automatically extracting narrative information such as characters and their roles in a given story from natural language text. In this paper, we focus on the character identification part of the system.

Entity Extraction. *Voz* uses the Stanford CoreNLP suite to segment the input text into sentences and to annotate them with the following information: part-of-speech (POS) tags (i.e., whether a word is a noun, a verb, etc.), syntactic parse trees (i.e., the syntactic structure of a sentence), typed dependency lists (i.e., relations between adjectives and the nouns they refer to, verbs and their subject and object complements, etc.) and lemmatization (i.e., removing inflection to obtain base words). Then *Voz* traverses each of the sentence parse trees looking for any "noun phrase" (NP) node, since NP nodes are candidate entities. For each NP node, *Voz* does the following. If the subtree contains nested clauses or phrases, or if the leaves of the subtree contain an enumeration (a conjunction or a list separator token), then *Voz* traverses its associated subtree recursively. Otherwise, if any leaf in the subtree is a noun, personal pronoun or possessive pronoun, the node is marked as an *entity*, and its subtree not explored any further. We do not use any form of coreference resolution during this stage. This results in a set of entities $E = \{e_1, ..., e_n\}$.

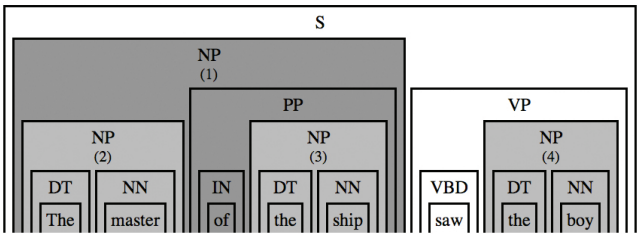

Figure 1: Syntactic parse of a sentence annotated by *Voz* after the entity extraction process.

Using this process, for example, *Voz* detected three entities, shaded in light gray in Figure 1, from the input sentence *"The master of the ship saw the boy."* After finding the compound NP "The master of the ship," (1) the system recursively detected two nested entities, "the master" (2) and "the ship" (3).

Feature-Vector Construction. For each entity $e_i \in E$, *Voz* computes a set of features. The features are computed from the parse tree of the sentence where the entity is found, the subtree representing the entity, the leaves of the subtree (i.e., word-level tokens with POS tags) and the dependency lists that contain a reference to any node in the entity's subtree. When looking at the word-level tokens, *Voz* queries knowledge bases such as WordNet, ConceptNet and some word lists (often referred in the NLP literature as dictionaries or gazetters). The list of specific features we compute is described in detail below. For now, it suffices to say that *Voz* computes a feature-vector of length m for each entity, where each feature is numeric and in the interval of $[0, 1]$.

Entity Classification. Given an entity e, *Voz* uses a case-based reasoning (CBR) approach to classify it into a set of classes S. For the purposes of this paper, $S = \{character, non\text{-}character\}$. As any CBR system, *Voz* contains a *case-base* $C = \{c_1, ..., c_l\}$, where each *case* $c_i = \langle e_i, s_i \rangle$ is composed of an entity e_i and a class $s_i \in S$. When classifying a new entity, the most similar instance to e from the case-base is selected, and the class of e is predicted as the that of the retrieved case. To determine the most similar case, we use a distance measure that we call the *Continuous Jaccard distance*.

The Jaccard index (Jaccard 1912) is a very well-known similarity function between two sets (A, B) defined as the "size of their intersection, divided by the size of their union":

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Most of the features computed by *Voz* represents whether an entity satisfies a given property or not (e.g., whether the entity if the subject of a verb or not). If the entity satisfies the property, then the feature has value 1, and if it does not, then the feature has value 0. Thus, we could see an entity e as a set of properties (that contains each property for which its corresponding attribute has value 1), and thus,

we should be able to apply the Jaccard index for assessing similarity between entities. However, some of the features are actually continuous (e.g., how similar is an entity to a given concept) and thus can take intermediate values like 0.5. Therefore, the standard formulation of the Jaccard index cannot be used directly. We hence generalized the notion of the Jaccard similarity by borrowing the notion of a t-norm form the field of Fuzzy Logic (Klement, Mesiar, and Pap 2000). Intuitively, a t-norm (triangular norm) is a function that generalizes the concepts of intersection in set theory and conjunction in logic. By interpreting the "intersection" and "union" in the Jaccard index as a t-norm and a t-conorm (the equivalent of the "union"), we replaced them by appropriate t-norms and t-conorms for numerical values: the *min* and *max* operators, resulting in the following measure, that we call *Continuous Jaccard*:

$$D_J(e_1, e_2) = 1 - \frac{\sum_{i=1}^{m} min(f_i(e_1), f_i(e_2))}{\sum_{i=1}^{m} max(f_i(e_1), f_i(e_2))}$$

where we write $f_i(e)$ to represent the value of the ith feature of the entity e. When the two entities are identical, D_J is 0. If they are completely disjoint, D_J is 1.

In the experiments reported in this paper, we compare the results obtained with this measure against two other standard distance measures:

- *Euclidean distance*: a standard euclidean distance between the feature vectors representing each entity:

$$D_E(e_1, e_2) = \sqrt{\sum_{i=1...m} (f_i(e_1) - f_i(e_2))^2}$$

- *Cosine distance* (Salton and Buckley 1988): a standard distance measure used in text retrieval:

$$D_C(e_1, e_2) = 1 - \frac{\sum_{i=1}^{m} f_i(e_1) f_i(e_2)}{\sqrt{\sum_{i=1}^{m} f_i(e_1)^2} \sqrt{\sum_{i=1}^{m} f_i(e_2)^2}}$$

Additionally, given that different features might contribute more or less to the classification of each entity, we experimented with two variants of each of the previous distance metrics: *standard* (as presented above) and *weighted*.

In the weighted versions of the similarity measures, we computed a numerical weight for each feature by using *Quinlan's Gain* (Quinlan 1986). In our previous work (Ontañón and Plaza 2012) in the context of distance measures, we observed that it achieved good results in computing feature weights. Weights are computed based on the cases in the case-base. For each feature f, the case-base is divided in two sets: C_1, with those cases where $f \leq 0.5$ and C_2, with those where $f > 0.5$. Then, Quinlan's Gain is computed as:

$$Q(f) = H(C) - \frac{H(C_1) \times |C_1| + H(C_2) \times |C_2|}{|C|}$$

where $H(C)$ represents the entropy of the set of cases C (with respect to the distribution of classes). We also evaluated other procedures for determining weights, such as using the mutual information. However, we obtained significantly worse performance using them.

The weighted versions of the three distance measures above result from multiplying the contribution of each feature to the similarity by the corresponding weight. Specifically, in each distance, each term in each of the summations (the sum in the Euclidean distance, the three sums in the Cosine distance, and the two sums in the Jaccard distance) is multiplied by the weights of the corresponding feature. The weighted Continuous Jaccard measure is thus defined as:

$$D_{wJ}(e_1, e_2) = 1 - \frac{\sum_{i=1}^{m} Q(f_i) min(f_i(e_1), f_i(e_2))}{\sum_{i=1}^{m} Q(f_i) max(f_i(e_1), f_i(e_2))}$$

Dataset

To test our approach, we use a collection of Russian folk tales translated into English as our dataset. The tales are either analyzed by Vladimir Propp (1973) or available in the Proppian Fairy Tale Markup Language collection[1]. As these have been translated by different authors these stories contain a variety of linguistic and narrative features and styles.

To reduce parsing issues when using Stanford CoreNLP we manually removed from the text: 1) dialogues, and 2) passages where the narrator addressed the reader directly (e.g., "If you think of it ..." where "you" refers to the reader). The original text in our dataset contains 384 sentences, as segmented by Stanford CoreNLP. After the edits, the text contains 269 sentences with an average length of 3.86 words, or 71.56 characters. The stories range from 7 to 48 sentences with an average of 28.75 sentences ($\sigma = 14.30$).

Specific to character identification, the text includes a range of different types of characters: humans, animals (e.g., a talking mouse) and anthropomorphic objects (e.g., a magical oven, a talking river). There are also fantastical creatures (e.g., goblins) and characters specific to the Russian folklore tradition (e.g., Morozko and Baba Yaga). We seek to identify these characters from the text without hand-coding specific knowledge such as knowing that "Morozko" is a traditional Russian character.

The dataset for our experiments consists of the entities extracted from the edited text. They are extracted using the entity extraction process described above. Specifically, there are 1122 entities, which were annotated as either characters or non-characters by hand for experimentation purposes[2].

Feature Set

Selecting the right features to describe an entity is key to character identification. Building upon the features proposed by (Calix et al. 2013), we included a set of additional features, showing a significant increase in accuracy. *Voz* currently extracts 193 different features per entity. Due to limitations of space, we will only describe the different types of features we use and how they are extracted instead

[1] Available: http://clover.slavic.pitt.edu/sam/propp/praxis/results.html

[2] Our complete dataset is available at: https://sites.google.com/site/josepvalls/home/voz

of providing a comprehensive list. The complete feature set is available online along with our dataset[3].

Linguistic Features

We included features at the linguistic level to help identify entities as characters or otherwise.

Sentence Parse Tree Features. *Voz* looks at the parse tree of the sentence where an entity is found and extracts features related to the nodes containing the entity, such as its depth and the presence of adverbial, adjectival or prepositional phrases. These features may indicate how relevant an entity is in a sentence or the semantic role it plays. There are 11 features in this category. For example, the *fromNodeS* feature captures whether the entity is found directly under a sentence node (S). *fromNodePP* describes whether the entity is found in a nested prepositional phrase node (PP).

Entity Parse Subtree Features. *Voz* traverses the subtree representing the entity and extracts features related to the types (i.e., syntactic labels) and number of nodes found indicating nested noun phrases. These features may indicate the presence of nested noun phrases, typically found in proper nouns and proper names composed of common nouns. There are 9 features in this category. For example, *hasNodeNP* captures that the entity has a nested noun phrase node (NP).

POS Tag Features. *Voz* enumerates the leaves of the entity subtree and extracts features related to the POS tags assigned to the word-level tokens. The features account for the presence of common nouns, proper nouns, pronouns, adjectives or existentials (e.g., "there") that may indicate specific types of entities. There are 26 features in this category. For example, *hasTokenJJ* captures that there is an adjective in the entity, and *hasTokenPRP$* captures that there is a possessive pronoun in the entity.

Dependency List Features. *Voz* enumerates the lists of dependencies and extracts several types of dependencies. A dependency, in this context, is a relation between two elements in a sentence (the *dependent* and the *governor*). For example, it can be the relation between a verb with its subject, direct object or indirect object, or the relation between a determiner and the noun it is referring to. In Figure 1, for instance, there is a dependency of type "direct-object" between "saw" (the governor) and "the boy" (the dependent).

For verb dependencies, we look at both active and passive voices for each identified verb in a sentence. Overall, *Voz* records if 1) an entity appears as a subject or object of any verb, 2) if it appears as subject or object of specific verbs (e.g., "have" or "tell") and 3) the similarity of the verb where an entity appears to some predefined clouds of concepts (e.g., one such cloud is "reward," "pay," "give"). Similarity between a verb and a cloud is computed similarly how the similarity between a noun and a cloud is computed, as described below. The features computed from these account for typological semantic roles for the verb cloud arguments. Typically for verbs like "talk" or "tell", the subject and direct object arguments are likely to be characters. For the verbs like "have" or "give", the object is less likely to be a character than the subject.

Voz also looks at other dependencies, specifically the relationships of nouns with prepositions, possessives and determiners. *Voz* computes features to identify when an entity has a determiner or wether an entity possesses something or is possessed by something else. These features indicate relationships between entities and an entity posessing something is more likely to be a character than an entity that is possessed by some other entity.

Overall, we use 3 features for determining if an entity appears as a subject of a verb and 6 to determine which argument of a verb an entity appears in, 62 features from verb dependencies for single verbs, 20 from verb clouds, 14 from prepositions, 4 from determiners, and 4 from other dependencies. For example, the *isVerbSubject* feature captures whether the entity appears as the subject of one (any) verb. *isVerbSubjectCloudMove* captures that the entity appears as the subject of a verb, this feature accounts for the similarity to a cloud of verbs like "move", "go" or "arrive." *depHeadPrepOf* captures that the entity is used as the head of a dependency with the preposition "of." *depHeadPoss* captures that the entity is the governor of a possessive dependency indicating that it is possessed by some other entity.

External Knowledge Features

Voz also uses external knowledge from WordNet, ConceptNet, and lists of words to compute features from words in the entity subtree.

WordNet-based Features. We defined several clouds of concepts, where each "concept" is a set of WordNet synsets. For example, one of these clouds is formed by synsets related to the concept of "youth" while another is related to the concept of "villainy". The similarity between the cloud and each noun (word with a noun POS tag) in the entity is computed, and the average value is the final value of the feature (if there are no nouns in the entity, then the feature takes value 0). To assess the similarity of a noun with a cloud of synsets, we use the measure proposed by Wu & Palmer (1994) to compute similarity between the noun and each of the synsets in the clouds. This measure returns a value between 0 and 1. Among the similarity values for each sysnet, the maximum value is used. There are 8 features in this category. An example feature is *hasWordnetAgeYoung*. This feature is based on the similarity of words in the entity to synsets related to "youth." The cloud contains adjectives like "young" and nouns like "baby", "child", "boy" and "girl". These features may indicate similarities with certain character archetypes and when building our clouds we used Propp's work as a reference. We defined 8 clouds with each containing between 3 and 17 synsets.

ConceptNet-based Features. Following the work of Calix et al. (2013), we query ConceptNet and look for some properties in the relationships of the returned concepts. We look at edges of certain types (e.g., "IsA" and "RelatedTo") connecting to some specific nodes (e.g., magic, wizard) to

[3]https://sites.google.com/site/josepvalls/home/voz

compute the 9 features in this category. For example, the feature *hasConceptnetHumanCapabilities* captures whether any nouns in the entity have ConceptNet relationships of the type "CapableOf" to concepts such as "laugh", "feel" or "love". There are 9 features and each checks between 3 and 9 edges.

Word List Features Our word lists are what are commonly referred to in the NLP literature as dictionaries or gazetters. The main difference between clouds and word lists is that *Voz* uses a similarity measure for clouds, and exact matching for word lists (i.e., if *Voz* finds a word from the entity inside of the given word list, the feature will have value 1, otherwise it will have value 0). There are 6 features from lists of nouns and 11 features from lists of other types words, with each feature defining its own word list and also the set of POS tags to filter the words that can be matched. We have a list of names including 2943 male and 5001 female names (an external list without modification based on birth records) and another with 306 common nouns including titles and professions. We have lists of words for identifying number (e.g., they, them), gender (e.g., he, she) and gender neutrality (e.g., another, other, this, that...). For example, the *hasWordInCommonNamesFemale* feature checks a word list of common nouns conceptually related to female gender or with female inflection like "witch", "nun", "lady" or "actress". The *hasWordInSingularThirdPersonGeneric* feature captures words identifying singular, third person objects such as "one", "another", "this" or "which." There are 7 additional lists and each has between 3 and 27 words.

Experimental Evaluation

To evaluate our approach we used the dataset presented above, containing 1122 entities. We hand-labelled these instances, obtaining 615 labelled as "Character" and 507 labeled as "Non-Character." Each entity is represented as a feature-vector with 193 features. The experiments presented in this section are designed to answer two main questions:

Q1: Which similarity measure is more effective in our case-based approach for character identification?

Q2: Among all the features we use, which ones are more effective?

To answer the first question, we evaluated the performance of the above described six distance measures: Euclidean, Cosine and Continuous Jaccard (S_E, S_C, S_J), and their corresponding weighted versions (S_{wE}, S_{wC}, and S_{wJ}). All the results presented in this section are the result of a cross validation procedure, which works as follows: given that the 1122 instances in our dataset come from 8 different stories, we split the 1122 instances into 8 different sets according to the story they come from. Then, when we try to predict the label for the entities in a given story, we only include in the case-base the instances coming from the other 7 stories.

Table 1 shows the performance of the 6 distance measures on our complete dataset. Performance is reported as accuracy, precision and recall. The first thing we can observe is that all the weighted variants of the distance measures outperform their non-weighted versions. For example, the weighted Euclidean distance (S_{wE}) can classify 88.41% of the instances correctly into characters/non-characters, whereas the non-weighted Euclidean distance (S_E) only classifies correctly 86.45%. Moreover, we can see that the best performance is achieved with our weighted-Continuous Jaccard distance measure (S_{wJ}), which correctly classifies 91.27% of the instances. This is also reflected in the precision and recall values.

Table 1: Performance of the different distance measures over the complete dataset (1122 instances) measured as classification accuracy (*Acc.*) and Precision/Recall (*PRec/Rec.*).

Distance	*Acc.*	*Prec./Rec.*
S_E	86.45%	0.91/0.83
S_{wE}	88.41%	0.92/0.86
S_C	87.08%	0.91/0.85
S_{wC}	89.22%	0.93/0.87
S_J	86.45%	0.91/0.84
S_{wJ}	91.27%	0.93/0.91

We compared our results against Stanford's *Named Entity Recognition* (NER), which achieved a precision of 0.98, but an extremely low recall, of 0.09, since it only recognizes, as *PERSON*, entities that have capitalized names. We also compared our results against standard machine learning classifiers, using the WEKA software. AdaBoost obtained precision and recall of 0.79/0.92. This unexpected low performance is due to the sensitivity of boosting classifiers to noise. Surprisingly, from the classifiers available in WEKA, J48 achieved the best performance with P/R scores of 0.92/0.91, slightly below that of S_{wJ}.

Moreover, we noticed that in our dataset, many characters are often referred to with personal pronouns (i.e., "he" or "she"), clearly indicating that they are characters (non-character entities, such as props or scenario elements are never referred to using these personal pronouns). So, in order to present our system with a bigger challenge, we repeated our experiments but removing all the instances that consisted of personal pronouns, excluding the pronoun "it" since it is used for both characters (anthropomorphic objects and animals) and non-characters (objects or settings). This left us with 886 instances. Results are reported in Table 2. As expected, performance decreased (specially for the non-weighted distance measures). However, our S_{wJ} distance measure was able to keep classification accuracy over 90%.

To answer our second question (Q2), we performed a set of experiments using our complete dataset (1122 instances) where we removed different sets of features from our dataset. Specifically Table 3 reports the performance of the S_{wJ} measure in the following scenarios: each row in Table 3 represents a different set of features (from the types of features described before); each row reports how many features are in each set, the performance of S_{wJ} when *only* using features of in the given set, and also when using all the features *except* the ones of in the given set. For example, the first row of Table 3 (*WordNet*) reports the classification accuracy that S_{wJ} obtains when only using the 8 features coming

Table 2: Performance of the different distance measures over the a reduced dataset removing all the "trivial" instances (886 instances) measured as classification accuracy (*Acc.*) and Precision/Recall (*Prec/Rec.*).

Distance	*Acc.*	*Prec./Rec.*
S_E	83.18%	0.86/0.72
S_{wE}	88.26%	0.90/0.82
S_C	83.74%	0.85/0.76
S_{wC}	88.71%	0.89/0.84
S_J	83.41%	0.86/0.74
S_{wJ}	90.18%	0.91/0.86

Table 3: Performance of the S_{wJ} distance measure with different feature subsets: *Acc. only* reports the accuracy only using features of a given type, and *Acc. all except* reports the accuracy using all the features except the ones of the given type. N reports the number of features of each type.

Feature Subset	N	*Acc. only*	*Acc. all except*
WordNet	8	81.11%	87.88%
Entity Parse Subtree	9	70.59%	90.81%
POS Tags	26	68.27%	91.09%
Sentence Parse Tree	11	66.84%	90.55%
Lists of Nouns	6	66.04%	88.15%
Verb Clouds	20	56.33%	91.00%
Lists of Words	11	54.72%	91.44%
Verb List	62	54.37%	91.18%
Prepositions	14	54.01%	91.09%
ConceptNet (Calix)	9	52.85%	90.46%
Verb Subject/Object	3	51.43%	91.89%
Other Dependencies	4	50.89%	90.91%
Verb Argument	6	50.45%	91.18%
Determiners	4	48.13%	91.53%

from the parse tree containing the entity (81.11%) and also when using all the features except those 11 (87.88%).

Analyzing the results, we can see that clearly the features that most contribute to the performance of our approach are those coming from WordNet: using only the 8 features coming from WordNet, S_{wJ} achieves a classification accuracy of 81.55%, which is remarkable. Other types of features that are very important are the *Lists of Words*, those coming from *Entity Parse Subtree*, and from the *Sentence Parse Tree*. Also, notice that there are some features that when removed performance actually increases (e.g., *Determiners*). Surprisingly, a feature that we hypothesized would be very helpful (whether an entity is the subject of a verb or not) does not actually help in the classification. Finally, there are some sets of features, such as those coming from ConceptNet (*ConceptNet (Calix)*), which contribute to the performance of the system, but are not enough for themselves to do any accurate classification of entities. Finally, notice that the most important set of features (*WordNet*) are continuous, justifying the need for our new Continuous Jaccard measure.

We performed an additional experiment removing the 3 sets of features that make the performance increase when removed, and we obtained a classification accuracy of 93.49% (a significant increase). In summary, we found that features coming from the parse tree, from the POS tags (*Nodes Within*), from WordNet, from several word lists were the most useful in classifying entities into characters and not characters. Features coming from specific verbs turn out to be very sparse for actually helping in the classification. Given the moderate size of our dataset, we believe that 93.49% is a very promising result, which could be improved by increasing the size of the dataset being used.

If we compare these results with work reported in the literature, Calix et al. (2013) report 84.3% accuracy using an Euclidean Distance, and 86.1% using Support Vector Machines. Performance is not strictly comparable, since they used a different dataset (with almost 5000 instances), but the numbers seem to indicate that our Continuous Jaccard measure, combined with the set of features we determined above, outperforms these results.

Conclusions

In this paper we presented a case-based approach for extracting narrative information, namely identifying characters, from folk tales in natural language text. We proposed a collection of features to perform this task, and a new similarity measure (Continuous Jaccard) that outperforms previous work on this problem. In order to evaluate our approach, we implemented it in *Voz*, a system that aims at automatically extracting narrative information from natural language stories. Specifically, given a story, *Voz* extracts a set of entities, computes a feature-vector for each one and then classifies those entities as characters and non-characters.

Our experimental results show that our approach significantly improves results wen compared to recent related work. Specifically, we achieve 93.49% accuracy on the task of classifying extracted entities as either characters and non-characters after filtering out a proper set of features. We also show that the set of features that are most important for the task of character identification are those exploiting the knowledge in WordNet. This shows that semantic information is key for determining what is a character.

The work presented in this paper is one step toward our long-term goal of automatically extracting high-level narrative information from natural language text. The improvements on our character identification method with respect to our previous work (Valls-Vargas, Ontañón, and Zhu 2013) will directly impact the performance of the overall *Voz* system, which aims at identifying not only characters, but also other high-level information such as their roles in the story.

As part of our future work, we plan on creating a larger dataset for better evaluation of our methods (which will be made publicly available). We also want to improve the methods described in this paper to go beyond classifying entities as characters/non-characters and also detect which ones are props, which ones refer to events, time anchors, or refer to the environment (i.e., the scenery surrounding a scene) or setting (e.g., the land where the story happens), in order to be able to fully classify each entity in a story along Chatman's narrative taxonomy (Chatman 1980).

References

Aamodt, A., and Plaza, E. 1994. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communications* 7(1):39–59.

Bringsjord, S., and Ferrucci, D. 1999. *Artificial Intelligence and Literary Creativity: Inside the Mind of BRUTUS, a Storytelling Machine*. Hillsdale, NJ, USA: L. Erlbaum Associates Inc.

Calix, R. A.; Javadpour, L.; Khazaeli, M.; and Knapp, G. M. 2013. Automatic Detection of Nominal Entities in Speech for Enriched Content Search. *The Twenty-Sixth International FLAIRS Conference* 190–195.

Chambers, N., and Jurafsky, D. 2008. Unsupervised Learning of Narrative Event Chains. *ACL* 94305(June):789–797.

Chambers, N., and Jurafsky, D. 2009. Unsupervised learning of narrative schemas and their participants. *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP* 2(August):602–610.

Chatman, S. B. 1980. *Story and discourse: Narrative structure in fiction and film*. Cornell University Press.

Elson, D. K. 2012. *Modeling Narrative Discourse*. Ph.D. Dissertation, Columbia University.

Finlayson, M. A. 2008. Collecting semantics in the wild: The story workbench. In *Proceedings of the AAAI Fall Symposium: Naturally-Inspired Artificial Intelligence*, 46–53.

Goyal, A.; Riloff, E.; and Daumé III, H. 2010. Automatically producing plot unit representations for narrative text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 77–86. Association for Computational Linguistics.

Jaccard, P. 1912. The distribution of the flora in the alpine zone. 1. *New phytologist* 11(2):37–50.

Klement, E. P.; Mesiar, R.; and Pap, E. 2000. *Triangular norms*. Springer.

Lehnert, W. G. 1981. Plot units and narrative summarization. *Cognitive Science* 5(4):293–331.

Ontañón, S., and Plaza, E. 2012. Similarity measures over refinement graphs. *Machine learning* 87(1):57–92.

Ontañón, S., and Zhu, J. 2011. On the role of domain knowledge in analogy-based story generation. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Two*, 1717–1722. AAAI Press.

Propp, V. 1973. *Morphology of the Folktale*, volume 9. University of Texas Press.

Quinlan, J. R. 1986. Induction of decision trees. *Machine learning* 1(1):81–106.

Regneri, M.; Koller, A.; Ruppenhofer, J.; and Pinkal, M. 2011. Learning Script Participants from Unlabeled Data. *RANLP*.

Salton, G., and Buckley, C. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management* 24(5):513–523.

Valls-Vargas, J.; Ontañón, S.; and Zhu, J. 2013. Toward character role assignment for natural language stories. In *Proceedings of the Ninth Artificial Intelligence and Interactive Digital Entertainment Conference*, 101–104.

Wolsey, L. A. 2000. Integer programming. *IIE Transactions* 32(273-285):2–58.

Wu, Z., and Palmer, M. 1994. Verbs semantics and lexical selection. In *Proceedings of the Thirty-Second annual meeting on Association for Computational Linguistics*, 133–138. Association for Computational Linguistics.