Narrative Hermeneutic Circle: Improving Character Role Identification from Natural Language Text via Feedback Loops

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

Abstract

While most natural language understanding systems rely on a pipeline-based architecture, certain human text interpretation methods are based on a cyclic process between the whole text and its parts: the hermeneutic circle. In the task of automatically identifying characters and their narrative roles, we propose a feedback-loop-based approach where the output of later modules of the pipeline is fed back to earlier ones. We analyze this approach using a corpus of 21 Russian folktales. Initial results show that feeding back high-level narrative information improves the performance of some NLP tasks.

1 Introduction

Computational narrative systems, especially story generators, require the narrative content to be encoded in some form of structured knowledge representation formalism. Currently, authoring content in ways that computer systems can reason about is mostly done by hand, a notoriously time-consuming task requiring expertise in both storytelling and knowledge engineering. We could alleviate this well-known "authorial bottleneck" problem if systems were able to directly take stories written in natural language and automatically extract information into the appropriate knowledge representation. More importantly, this would make it easier for computational narrative researchers and practitioners to utilize the vast amount of existing literary work, making use of its narrative content as well as storytelling techniques.

Except for a few recent exceptions, automatically extracting high-level narrative information from unannotated text has not received a lot of attention. Here we focus on the problem of automatically identifying characters and their roles from unannotated Russian folktales translated into English. In this domain, *characters* are sentient beings such as persons, anthropomorphic animals and other magical beings (e.g., a dragon or a talking oven). We use the roles identified in Vladimir Propp's narrative theory of Russian folktales [1973]. Consider the following excerpt:

The sister walked and walked underground and saw a little hut. [...] Inside sat a lovely maiden embroidering a towel with silver and gold. She received her guest with kindness. [...] when the hostess knew that her mother was about to come she turned her guest into a needle [...].

Based on Propp's theory, the "maiden" *character* fulfills the specific function of the "helper" role in the plot through a specific "sphere of action" including key actions related to interrogation and entreaty.

Elsewhere we presented a framework to identify character roles in this domain [Valls-Vargas *et al.*, 2013; 2014]. Our experiments indicated that one of the bottlenecks is coreference and anaphora resolution (we will refer to both as coreference resolution for simplicity). Notice the "helper" character (the referent) in the excerpt above is referred to with several referring expressions: "maiden", "she" and "hostess". The difficulty of identifying all the expressions of the same referent, a well-known open problem in NLP, significantly lowered our system's performance. Moreover, we observed that some coreference resolutions could be resolved if earlier components were given the information of character roles derived later in the process. For example, knowing that that both the "maiden" and the "hostess" play the role of the helper is a hint that they are referring to the same referent.

Our observation is aligned with certain methods in humanistic interpretation of complex text. When making sense of the text, humanities scholars tend to move back and forth between its individual parts and the larger whole. They believe that our understanding of the text as a whole is hinged on our understanding of the individual parts, which in turn is shaped by how we understand the larger whole. This circular process of understanding text is the most basic form of *the hermeneutic circle* [Spinoza, 2001]. This concept was further developed into a broader philosophical level and directly influenced early NLP research [Mallery *et al.*, 1986]. In this paper, the hermeneutic circle refers to the classical text interpretation methodology whereby scholars are trained to iteratively shifting their attention between the parts and the entirety of the text in order to refine and deepen their understanding. By borrowing this terminology, we intend to call attention to the inter-dependency between local linguistic-level information (e.g., coreference) and the global narrative-level information (roles) as well as the iterative understanding process, neither of which has been sufficiently explored.

In this paper, we present our first step toward a feedback-loop-based narrative text understanding approach for the task of character role identification, incorporated in *Voz*, our fully automatic narrative extraction system. Compared to our prior work, this paper addresses two main questions: 1) how to incorporate information fed back from later stages of the NLU pipeline, and 2) how the feedback loop impacts the performance of the system on a corpus of 21 folktales. Although we use the domain of Russian folktales, we believe that our approach can be generalized to other types of stories where character roles are closely tied to their actions. Our new experiments show that feedback loops can indeed increase the performance of some modules (coreference resolution), although this does not translate into increased performance of other modules in the system (e.g., role identification).

2 Related Work

Following early work in narrative understanding through Schankian knowledge structures and cognitive processes [Schank and Abelson, 1977; Cox and Ram, 1992], advances in NLP and related disciplines have recently led to a renewed interest in extracting and modeling narrative elements from text. Character identification, related to named entity recognition and nominal actor detection, is a crucial step in this process. 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 uses 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. Chambers and Jurafsky [2008] proposed using unsupervised induction to learn what they called "narrative event chains" from raw newswire text. Regneri et al [2011] worked on the specific task of identifying matching participants in given scripts in natural language text using semantic (WordNet) and structural similarities with Integer Linear Programming [Wolsey, 1998]. More recently, Calix et al. [2013] proposed an approach for detecting characters in spoken stories. Based on features in the transcribed textual content using WordNet and speech patterns (e.g., pitch), their system detects characters through supervised learning techniques. Compared to these systems, our work focuses on how the extraction of high-level narrative information (characters and their roles) can help with lower-level NLP tasks (coreference) via feedback loops.

Recent systems such as *Story Workbench* [Finlayson, 2012] and *Scheherazade* [Elson, 2012] employ a mixed-initiative approach to annotating text with narrative information via semi-automatic assistance. These systems are flexible and enable the annotation of high level narrative information but the process can still be time-consuming and requires the participation of human experts.

Pipelined architectures are used in many NLP tasks [Clarke *et al.*, 2012], specially, natural language understanding and generation [Reiter and Dale, 2000]. Although the flexible modularity provided by pipelines can be very effective, it is not best suited for all problems. Several researchers have independently shown the advantages of non-linear approaches such as *blackboard architectures* [Erman *et al.*, 1980], global optimization [Marciniak and Strube, 2005; Roth and Yih, 2004] (where the outputs of the individual modules are considered by a global optimization process, such as Linear Programming), or context awareness [Lee *et al.*, 2011]. Additionally, backtracking and re-interpretation are known to be a key part in computational understanding of humor [Attardo and Raskin, 1991] and figurative language [Gibbs, 1994]. Finally, although the use of feedback loops on pipelined architectures has been suggested before [Samuelsson *et al.*, 2008], to the best of our knowledge, there has been very little existing work on implementing iterative architectures for NLP-based information extraction.

3 Problem Statement

Propp categorized characters in Russian folktales into several basic functional roles or character functions (*roles* from now on): Hero, Villain, Dispatcher, Donor, (Magical) Helper, Sought-for-person, and False Hero. Each role fulfills specific narrative functions and performs its specific "sphere of action." Our goal is to capture this structural regularity and identify roles regardless of the specificity of how characters and actions are constructed at the story and discourse levels.

The problem we address is: given an unannotated story, how to identify the set of characters $\{a_1, ..., a_n\}$ from the referring expressions in the text, and how to identify, from a given set of roles $\{r_1, ..., r_m\}$, which is the most likely narrative role that each character portrays.

Our hypothesis is that the feedback loop between high-level role information and low-level linguistic information can improve the overall system's performance compared to using a pipeline-based architecture.

4 Feedback loop-based Role Identification

In this section we describe our fully-automated system called *Voz*. Figure 1 illustrates the information workflow in *Voz* and the feedback loop. Below, we briefly describe each module, with the emphasis on the feedback loop and how the coreference resolution module is designed in order to take into account information produced by later modules in the pipeline. A more in-depth description of the other modules in the system can be found in [Valls-Vargas *et al.*, 2014].

Voz uses the off-the-shelf Stanford CoreNLP suite [Manning *et al.*, 2014], to segment the input text into sentences and to annotate them with the following information: part-of-speech (POS) tags (i.e., noun, verb.), syntactic parse trees, typed dependency lists and lemmatization. *Voz* also retrieves the output of the Stanford Deterministic Coreference Resolution system [Lee *et al.*, 2013].

Then, a custom component extracts the referring expressions ("mentions" from now on) from the text. A mention is a phrase that refers to a specific entity (i.e., the referent) in the story. For example "the sister", "she", or "her guest" are separate mentions to the same character. *Voz* traverses the sentences' parse trees looking for "noun phrase" (NP) nodes, traversing the subtrees of these nodes in case there is an enumeration (e.g., a conjunction). For each mention, *Voz* computes a vector of features based on the following information: the parse tree of the sentence where the mention is found, the

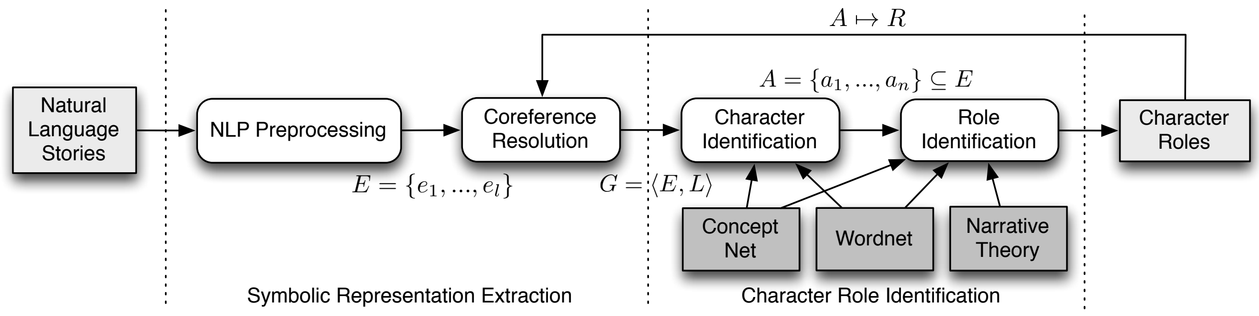

Figure 1: Architecture of the *Voz* system. Arrows represent the information workflow.

subtree representing the mention, 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 mention's subtree. When looking at the word-level tokens, *Voz* queries knowledge bases such as WordNet and ConceptNet. The feature vectors encode both local information and the "sphere of action" described in our previous work [Valls-Vargas *et al.*, 2014]. The output is a set of mentions $E = \{e_1, ..., e_n\}$, represented as feature vectors. In the experiments reported in this paper, each mention consists of 332 numeric features[1] with values are in the interval of $[0, 1]$.

Coreference Resolution. The goal of coreference resolution is to group the mentions E into a set of *coreference groups*, where each coreference group is a set of mentions that refers to the same character or entity in the story. We designed a coreference resolution approach based on aggregating the output of a number of base coreference resolution methods. This aggregation approach allows the feedback loop to take place, since the information fed back from role identification is used as one of the base methods to be integrated.

To represent the results from multiple base coreference resolution methods, the output of each of those methods is encoded as a *coreference preference matrix* (CP matrix). A CP matrix m is an $l \times l$ matrix. $m(i, j) = 1$ means that the corresponding method believes that e_i and e_j are expressions for the same referent; $m(i, j) = -1$ means the method believes they are not; and $m(i, j) = 0$ means the method is unsure (intermediate values are allowed, to indicate degrees of confidence). These matrices are then aggregated cell by cell into a joint *coreference assignment* using the method described shortly below. Figure 2 illustrates the entire process.

Currently *Voz* uses six coreference resolution methods, represented as six CP matrices:

- m_1 is derived from the output of the *Stanford coreference resolution system*, and only yields -1s and 1s.
- m_2 computes coreference by assigning a 1 to a pair of mentions when there are matching common nouns and proper nouns in the leaves of the extracted parse trees for each mention. m_2 only contains 0s and 1s. m_2 intends to consolidate mentions that use similar referring expressions (e.g., "a girl" and "the girl").

[1]The dataset and feature descriptions are available at: https://sites.google.com/site/josepvalls/home/voz

- m_3 captures gender or number (singular vs plural) disagreement between pairs of nouns and pronouns (e.g., "girl" and "him"). The matrix contains a -1 for those pairs of mentions for which a disagreement is detected, and 0 for all others. 15 lexical and syntactic features are used to infer the gender and number of each mention.
- m_4 computes coreference by considering 8 lexical and syntactic features and computing the similarity between these features in the two mentions. This captures semantic similarities between mentions (e.g., "girl" and "sister"), and contains values in the interval $[0, 1]$.
- m_5 computes coreference based on the roles and characters fed back via the feedback loop described below. It assigns a 1 to a pair of mentions when they correspond to characters with the same role (except if they have the *other* role), and 0 otherwise. m_5 intends to consolidate mentions to the same character, assuming not too many characters share the same role.
- m_6 is similar to m_5, but assigns a 0 to a pair of mentions when they correspond to characters with the same role (except if they have the *other* role), and -1 otherwise. The reason for having both m_5 and m_6, even if they capture the same information, is for being able to assign different weights to the 1s and -1s in these matrices in the aggregation process described below.

In the first iteration of *Voz* , only m_1, m_2, m_3 and m_4 are available. In subsequent iterations, the character and role labels predicted in the previous iteration are used to generate m_5 and m_6. Note that coreference can be understood as a graph $G = \langle A, L\rangle$ where the A is the set of mentions, and $L \in A \times A$ contains those pairs of mentions that refer to the same character or entity. Cliques (fully connected sets of nodes) in this graph represent coreference groups. *Voz* currently does not consider the directionality of the graph and therefore split antecedents (i.e., plurals) are ignored (split antecedents represent only 142 instances out of the 2,781 character mentions annotated in our dataset).

The CP matrices are then aggregated in the following way. First, a new matrix m_{merged} is generated as:

$$m_{merged}(i, j) = \begin{cases} 1 & \text{if } 0 \leq \sum_k w_i \times m_k(i, j) \\ 0 & \text{otherwise} \end{cases}$$

However, this merged matrix might not be fully consistent (e.g., if $m_{merged}(1, 2) = 1$ and $m_{merged}(2, 3) = 1$, then

$$w_1 \underbrace{\begin{array}{c|cccc} & e_1 & e_2 & e_3 & e_4 \\ \hline e_1 & 1 & 1 & 0 & -1 \\ e_2 & 1 & 1 & 0 & 0 \\ e_3 & 0 & 0 & 1 & 0 \\ e_4 & -1 & 0 & 0 & 1 \end{array}}_{m_1} + w_2 \underbrace{\begin{array}{c|cccc} & e_1 & e_2 & e_3 & e_4 \\ \hline e_1 & 1 & 0 & 1 & 1 \\ e_2 & 0 & 1 & 0 & 0 \\ e_3 & 1 & 0 & 1 & 0 \\ e_4 & 1 & 0 & 0 & 1 \end{array}}_{m_2} + \ldots + w_6 \underbrace{\begin{array}{c|cccc} & e_1 & e_2 & e_3 & e_4 \\ \hline e_1 & 0 & -1 & 0 & -1 \\ e_2 & -1 & 0 & 0 & 0 \\ e_3 & 0 & 0 & 0 & 0 \\ e_4 & -1 & 0 & 0 & 0 \end{array}}_{m_6} \longrightarrow \underbrace{\begin{array}{c|cccc} & e_1 & e_2 & e_3 & e_4 \\ \hline e_1 & 1 & & 1 & \\ e_2 & & 1 & & \\ e_3 & 1 & & 1 & \\ e_4 & & & & 1 \end{array}}_{G = \langle E, L \rangle}$$

Figure 2: Example pair-wise coreference preferences (CP) matrices. Independent weights are assigned to each of the matrices, which are then aggregated to determine pair-wise coreference grouping between mentions.

$m_{merged}(1,3)$ should be also 1). This matrix is thus made consistent by adding the missing 1s, and turned into a graph G, where each clique represents a coreference group.

Character Identification. Given a mention e, *Voz* uses a case-based reasoning (CBR) approach to classify e into a set of classes $S = \{\mathit{character}, \mathit{non\text{-}character}\}$. As a 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 a mention e_i (represented by the feature vector described above) and a class $s_i \in S$. When classifying a new mention, the most similar instance to e from the case-base is selected, and the class of e is predicted as that of the retrieved case. To determine the most similar case, we use a *Weighted Continuous Jaccard Distance*, shown in our previous work to outperform other common measures in this task [Valls-Vargas *et al.*, 2014].

Once all the mentions have been classified, the output of the coreference resolution is used to refine the results. Given a mention $e \in E$, we identify its *coreference group* $\mathit{coref}(e)$, i.e., all the other mentions that are linked to e in the coreference graph G. Then, the class assigned to e is replaced by the majority class in the coreference group $\mathit{coref}(e)$. For example, if two mentions in $\mathit{coref}(e)$ were labeled as $\mathit{character}$ and only one as $\mathit{non\text{-}character}$, then e will be labeled as $\mathit{character}$. The output is the set $A = \{a_1, ..., a_n\} \subseteq E$ of mentions that were classified as $\mathit{character}$.

Role Identification. Similar to the previous module, given a character $a \in A$, *Voz* uses a case base to predict a class for each character from a set of classes representing roles. The same *Weighted Continuous Jaccard Distance* is used for this prediction task. As in the previous process, the coreference group is used to perform a voting process, and the role of each character is assigned as the role assigned to the majority of the mentions in its coreference group.

Feedback Loop. *Voz* implements an architecture inspired by the idea of hermeneutic circles by incorporating a feedback loop from the output of the system to the input of the coreference resolution process, as shown in Figure 1. Thus, the execution of *Voz* requires not just one run through the pipeline, but several iterations. As described above, the coreference resolution process uses several base coreference resolution methods, whose output is aggregated to produce the final prediction. After character and role identification are complete, this information is fed back to the coreference resolution module, resulting in matrices m_5 and m_6, and starting a new iteration of the execution of *Voz*.

The Stanford coreference resolution system, which is the main source of coreference resolution in *Voz* , tends to be conservative and generates small coreference groups. Based on the assumption that some of the roles are only portrayed by one or at most a very small set of characters (e.g., "hero", or "villain"), it is our hypothesis that the additional m_5 and m_6 matrices should significantly improve the coreference resolution process as we explore in our evaluation below.

5 Experiments

This section reports a series of experiments to test: 1) the performance of the system as a pipeline in identifying characters and roles, 2) the effect of the feedback loop in the performance of the different modules of the system, and 3) the effect of varying the weight applied to the CP matrices that come form the feedback loop during coreference resolution. Before presenting such experiments, we describe our dataset.

5.1 Dataset

Our dataset contains 21 Russian folk stories translated to English text, collected by other researchers [Malec, 2010; Finlayson, 2012]. To reduce NLP preprocessing issues at the discourse level, we manually removed quoted and direct speech (i.e., dialogues and passages where the narrator addressed the reader directly). Our edited dataset contains 914 sentences. The stories range from 14 to 69 sentences ($\mu = 43.52$ sentences, $\sigma = 14.47$).

Despite their relatively short lengths, understanding these stories requires significant commonsense knowledge and contextual inference. For example in our Story S_{13}, there are two young female characters, who are both referred as "she", "daughter" and "maiden" throughout the story while they fulfill different narrative roles.

To quantify the performance of separate modules in *Voz*, we hand-annotated different aspects of the dataset as the ground truth. Specifically, we annotated all noun phrases (NP) representing referring expressions (mentions). There are 4791 annotated mentions, 2781 of which are characters.

Coreference groups are annotated for characters and mentions to groups of characters. Then the characters were annotated with character role labels described in the 31 Proppian functions and subfunctions [Propp, 1973]. We merged the roles of *Donor* and *Helper* since, in our dataset, they mostly correspond to the same character. We also merged *Dispatcher* with several other minor roles into an *Other* role. This resulted in six role labels. Additionally, we created an even coarser classification including only *Hero*, *Villain* and everything else as *Other*. The annotations were performed by two researchers independently and disagreement resolved manually by consensus. The experiments below are reported using both the set of six, and the set of three roles.

5.2 Experiment 1: Pipeline Performance

This experiment assesses the system's performance before closing the feedback loop. The weights used for the different CP matrices during coreference resolution were $w_1 = 1.0$, $w_2 = 1.1$, $w_3 = 10$ and $w_4 = 0.9$ (specific values are not important, and the only important aspect is that $w_2 > w_1$, and that w_3 is sufficiently large as for canceling all other weights out). Our dataset only contains coreference annotations for characters, and thus we only evaluate the performance of coreference resolution on the 2781 mentions that are annotated as characters. Our coreference process groups those 2781 mentions into to 1359 coreference groups. To evaluate the accuracy of the process, based on our ground truth annotations, we compute the average number of different characters found per coreference group (C/Gr), and the average number of different groups a single character is spread across (Gr/C). Perfect coreference would score C/Gr = 1.00, and Gr/C = 1.00 meaning that each group only contains mentions to one character and a character is mentioned in only one group respectively. Errors in coreference resolution will make these values higher. *Voz* obtains C/Gr = 1.07, and Gr/C = 6.00. This means that while *Voz* is relatively good at separating mentions from different characters, it does not work so well at merging different mentions of the same character.

Table 1 shows the precision, recall, f-measure and accuracy of the predictions for character and role identification, showing both the performance of these modules before and after voting (recall that these modules predict characters and roles for each individual mention, and then a voting process is performed among all the mentions of each coreference group). Precision, recall and f-measure values were computed for each class (*character/non-character* and each of the roles) comparing the predicted and annotated value for each mention. The reported values are averages weighted by the total number of annotated mentions for each class in the ground truth. Accuracy corresponds to the proportion of properly classified mentions of the total number of annotated mentions. Role predictions are shown both when trying to predict all six roles (Roles 6), and when trying to predict the abstracted set of three roles (Roles 3). When predicting roles, characters can be assigned any of the six or three role labels, and non-characters are assigned a special label (*non-character*). As we can see, character prediction is very accurate (reaching an accuracy of 0.866 before voting and 0.873 after voting). Role prediction accuracy is 0.645 for the six role prediction, and slightly higher for the three role prediction (both benefiting slightly from voting via coreference information). These numbers might seem low, but are significantly higher than those previously reported for similar datasets [Valls-Vargas *et al.*, 2013].

Moreover, since the dataset is biased towards *non-characters*, the role prediction accuracies are biased. So, the bottom two rows of the table include the performance when only considering those mentions classified as characters by *Voz*. The accuracy in predicting roles for the six role prediction is 0.438, which might appear low, but is again higher than previously reported, and way above a baseline predictor that just predicts a role at random. Table 2 reports the confusion matrix for this process (after voting) using six roles.

Table 1: Performance of the character and role identification processes over the 4791 extracted mentions.

	P	R	f	Acc
Characters (no voting)	0.844	0.877	0.860	0.866
Roles 6 (no voting)	0.639	0.618	0.624	0.639
Roles 3 (no voting)	0.664	0.654	0.655	0.664
Characters (voting)	0.874	0.883	0.878	0.882
Roles 6 (voting)	0.681	0.642	0.658	0.645
Roles 3 (voting)	0.667	0.657	0.658	0.667
Roles 6 (voting, filt.)	0.438	0.357	0.386	0.438
Roles 3 (voting, filt.)	0.484	0.430	0.448	0.484

Table 2: Confusion matrix for the six character roles as predicted by the first iteration of *Voz*. The roles are: False Hero (FH), Hero (H), Other (O), Sought-for-person (SP), Donor/Helper (D/H), and Villain, plus an additional *Non-character (N/C) role,*.

	N/C.	FH	H	O	SP	D/H	V
N/C	2087	0	77	179	1	0	9
FH	0	0	5	0	0	1	3
H	102	1	362	271	3	5	106
O	124	2	160	432	20	10	50
SP	10	0	10	15	28	0	13
D/H	69	0	46	55	0	2	17
V	109	0	83	116	23	6	179

5.3 Experiment 2: Feedback Loop Performance

Experiment 2 assesses the efficiency of the feedback loop described in Section 4. Now the coreference resolution process will have access to all six CP matrices which results in a different coreference assignment that will be used in the next iteration, yielding a different set of characters and potentially different roles for each. Weights for the first four matrices are the same as in the previous experiments, $w_5 = 0.9$, and $w_6 = 10.0$. The rationale for these weights is that $w_4 + w_5 > w_1$, thus when both m_4 and m_5 predict that two mentions should be grouped, that can overrule the decision made by the Stanford parser (m_1). w_6 is just set to a very large weight, since mentions with different roles cannot refer to the same character. Specific values for the weights do not have a strong impact as long as these relationships are kept.

Tables 3 and 4 show the obtained results when trying to predict the six-role annotations and the three-role annota-

Table 3: Performance of the first three iterations of *Voz* using 6 role classes. Last two rows display the theoretical upper bound using the ground truth and a random baseline.

	Coref. Resolution			Char.	Role	
	$\|Gr\|$	C/Gr	Gr/C	Acc	Acc	Acc (F)
It 1	1359	1.07	6.00	0.87	0.64	0.44
It 2	888	1.21	4.36	0.87	0.64	0.44
It 3	888	1.21	4.36	0.87	0.64	0.44
GT 6	642	1.24	2.90	0.88	0.73	0.46
Rnd.	642	1.93	3.11	-	-	-

Table 4: Performance of the first three iterations of *Voz* using 3 role classes. Last two rows display the theoretical upper bound using the ground truth and a random baseline.

	Coref. Resolution			Char.	Role	
	$\|Gr\|$	C/Gr	Gr/C	Acc	Acc	Acc (F)
It 1	1359	1.07	6.00	0.87	0.67	0.48
It 2	952	1.18	4.56	0.87	0.67	0.48
It 3	952	1.18	4.56	0.87	0.67	0.48
GT 3	788	1.18	3.44	0.88	0.73	0.47
Rnd	788	1.75	3.21	-	-	-

tions respectively. Each of the first three rows of the tables shows the performance of the system during the first three iterations: "It 1" just runs *Voz* as a pipeline, "It 2" runs the feedback loop once, and "It 3" runs it twice. Accuracy for character and role identification is reported after voting, and "Acc (F)" is the accuracy when measured only for those mentions classified as characters. We observed how the number of coreference groups decreases over the first iterations and remains stable afterwards. From the initial 2781 mentions, the Stanford deterministic coreference system yields 1359 groups. Considering results in Table 3, a second iteration, using the six CP matrices further reduces the number of coreference groups to 888 while improving coreference performance. The average groups per character is greatly reduced to Gr/C = 4.36, while the average characters per group only grows slightly to C/Gr = 1.21. Although still far from the ideal 173 coreference groups in the ground truth, the feedback loop clearly increases the performance of coreference resolution. Moreover, we observed no significant impact of the feedback loop on character and role predictions, which improved only marginally (an improvement smaller than the precision shown in the tables). The output of the system stabilized in the third iteration which obtained the same exact results as the second.

Moreover, in order to calculate the upper-bound for the improvement in performance achieved by the feedback loop, we experimented by feeding back the ground truth for role labels. This is shown in the row labeled "GT 6" in Table 3, showing an even better coreference resolution performance: reducing Gr/C further to 2.90 while C/Gr increased only slightly to 1.24, and reducing the number of groups from 1359 to 642, which is a significant improvement. Character and role predictions also improved (from 0.87 to 0.88 for characters and from 0.67 to 0.73 for roles), showing further potential of our feedback loop approach. Finally, in order to validate that the role information is useful, we tried joining coreference groups randomly (starting with the coreference groups obtained in iteration 1) until the same number of groups (642) were reached. This resulted in really worsening results, further indicating that role information is indeed useful in improving coreference resolution. Table 4 shows similar trends in the three-role prediction setting.

5.4 Experiment 3: CP Matrix Weights

Finally, we experimented with different values for weight w_5, which is the weight given to the CP matrix generated from the fed-back roles. This has the expected effect: higher w_5 results in role predictions having a stronger effect, and thus result in a lower coreference group count. Table 5 shows the results of different values for w_5 after the second iteration using the 6 roles. As the table shows, the main effect of increasing w_5 is reducing the number of coreference groups. From 1359 (with $w_5 = 0.1$) to 879 (with $w_5 = 1.5$) while only increasing the ratio of characters per group (C/Gr) slightly (from 1.17 to 1.22). Effects on character and role prediction are smaller than the precision shown in the table.

Table 5: Performance comparison with different weights for m_5, built form the feed back role information.

	Coref. Resolution			Char.	Role
	$\|Gr\|$	C/Gr	Gr/C	Acc.	Acc.
$w_5 = 0.1$	933	1.17	4.46	0.87	0.65
$w_5 = 0.25$	894	1.20	4.36	0.87	0.65
$w_5 = 0.5$	890	1.21	4.36	0.87	0.65
$w_5 = 0.9$	888	1.21	4.36	0.87	0.65
$w_5 = 1.5$	879	1.22	4.34	0.87	0.65

6 Conclusions and Future Work

This paper has presented one step toward achieving the goal of automatically extracting narrative information from natural language text. Our long term goal is to create computational narrative systems, and in particular story generation systems that feed directly from stories written in natural language. We presented an experimental evaluation of the effectiveness of a feedback loop inspired in the *hermeneutic circle* to improve the performance of *Voz*, a system to identify characters and their roles in natural language stories. We reported an empirical evaluation on a corpus of 21 Russian stories using Propp's narrative theory [1973] to annotate characters and roles.

Our experiments confirm that the idea of feedback loops can increase the performance of some of the modules of the system, specifically coreference resolution, but not others, such as character or role identification. The feedback loop introduced in the NLP pipeline of our system has been demonstrated to be particularly successful in reducing the number of coreference groups in the output of the system.

Feeding back role information to coreference resolution is only our first step. As part of our future work, we plan to explore the possibilities of feeding back additional information (such as Proppian functions) to additional modules besides coreference resolution. We are particularly interested in improving the performance of semantic role labeling, which is the next bottleneck in the performance of our system. We also plan on generalize our approach to other narrative domains by incorporating Campbell's monomyth theory [2008].

Acknowledgments

Special thanks to Mark A. Finlayson for kindly sharing his dataset of translated and annotated Russian stories.

References

[Attardo and Raskin, 1991] Salvatore Attardo and Victor Raskin. Script theory revis(it)ed: Joke similarity and joke representation model. *Humor: International Journal of Humor Research*, 1991.

[Calix *et al.*, 2013] Ricardo A. Calix, Leili Javadpour, Mehdi Khazaeli, and Gerald M. Knapp. Automatic Detection of Nominal Entities in Speech for Enriched Content Search. In *Proceedings of FLAIRS 2013*, pages 190–195, 2013.

[Campbell, 2008] Joseph Campbell. *The hero with a thousand faces*, volume 17. New World Library, 2008.

[Chambers and Jurafsky, 2008] Nathanael Chambers and Dan Jurafsky. Unsupervised Learning of Narrative Event Chains. *ACL*, 94305(June):789–797, 2008.

[Clarke *et al.*, 2012] James Clarke, Vivek Srikumar, Mark Sammons, and Dan Roth. An nlp curator (or: How i learned to stop worrying and love nlp pipelines). In *LREC*, pages 3276–3283, 2012.

[Cox and Ram, 1992] Michael T. Cox and Ashwin Ram. Multistrategy learning with introspective meta-explanations. In *Proceedings of the ninth international workshop on Machine learning*, pages 123–128. Morgan Kaufmann Publishers Inc., 1992.

[Elson, 2012] David K. Elson. *Modeling Narrative Discourse*. PhD thesis, Columbia University, 2012.

[Erman *et al.*, 1980] Lee D Erman, Frederick Hayes-Roth, Victor R Lesser, and D Raj Reddy. The hearsay-ii speech-understanding system: Integrating knowledge to resolve uncertainty. *ACM Computing Surveys (CSUR)*, 12(2):213–253, 1980.

[Finlayson, 2012] Mark A. Finlayson. *Learning narrative structure from annotated folktales*. PhD thesis, Massachusetts Institute of Technology, 2012.

[Gibbs, 1994] Raymond W Gibbs. *The poetics of mind: Figurative thought, language, and understanding*. Cambridge University Press, 1994.

[Goyal *et al.*, 2010] Amit Goyal, Ellen Riloff, and Hal Daumé, III. Automatically producing plot unit representations for narrative text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 77–86, Stroudsburg, PA, USA, 2010.

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

[Lee *et al.*, 2013] Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Deterministic coreference resolution based on entity-centric, precision-ranked rules. 2013.

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

[Malec, 2010] Scott Malec. Autopropp: Toward the automatic markup, classification, and annotation of russian magic tales. In *Proceedings of the First International AMICUS Workshop on Automated Motif Discovery in Cultural Heritage and Scientific Communication Texts*, pages 112–115, 2010.

[Mallery *et al.*, 1986] John C Mallery, Roger Hurwitz, and Gavan Duffy. Hermeneutics: from textual explication to computer understanding? 1986.

[Manning *et al.*, 2014] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, 2014.

[Marciniak and Strube, 2005] Tomasz Marciniak and Michael Strube. Beyond the pipeline: Discrete optimization in nlp. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 136–143. ACL, 2005.

[Propp, 1973] Vladimir Propp. *Morphology of the Folktale*. University of Texas Press, 1973.

[Regneri *et al.*, 2011] Michaela Regneri, Alexander Koller, Josef Ruppenhofer, and Manfred Pinkal. Learning Script Participants from Unlabeled Data. *RANLP*, 2011.

[Reiter and Dale, 2000] Ehud Reiter and Robert Dale. *Building Natural Language Generation Systems*. 2000.

[Roth and Yih, 2004] Dan Roth and Wen-tau Yih. A linear programming formulation for global inference in natural language tasks. Technical report, DTIC Document, 2004.

[Samuelsson *et al.*, 2008] Yvonne Samuelsson, Oscar Täckström, Sumithra Velupillai, Johan Eklund, Mark Fišel, and Markus Saers. Mixing and blending syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 248–252. Association for Computational Linguistics, 2008.

[Schank and Abelson, 1977] Roger C. Schank and Robert Abelson. Scripts, goals, plans, and understanding, 1977.

[Spinoza, 2001] Benedictus de Spinoza. *Theological-political Treatise*. Hackett Publishing House, Indianapolis, gebhardt edition, 2001.

[Valls-Vargas *et al.*, 2013] Josep Valls-Vargas, Santiago Ontañón, and Jichen Zhu. Toward character role assignment for natural language stories. In *Proceedings of the Ninth Artificial Intelligence and Interactive Digital Entertainment Conference*, pages 101–104, 2013.

[Valls-Vargas *et al.*, 2014] Josep Valls-Vargas, Santiago Ontañón, and Jichen Zhu. Toward automatic character identification in unannotated narrative text. In *Proceedings of the Seventh Workshop in Intelligent Narrative Technologies (INT 7)*, 2014.

[Wolsey, 1998] Laurence A Wolsey. *Integer programming*, volume 42. Wiley New York, 1998.