Fairy Tales for Grown Ups

7th June 2005

Ori Garin	Ganna Lobanova	Stijn van Balen
ogarin@student.uva.nl	G.Lobanova1@student.uva.nl	sbalen@student.uva.nl

1 Abstract

Speech synthesizers provide the computer with the possibility to speak. They are developed to produce natural-sounding output. However, the ability of speech synthesizers to mimic human speech is limited in many ways. For example, speech synthesizers don't understand what they say, and as a result, sometimes they do not use the right style or do not emphasize useful information.

This paper presents an insight on our attempt to provide a synthesizer with more understanding of its input in a shallow approach. The goals of our project can be summarized as follows:

- to achieve a more natural like output of children stories by
  - identifying the speech quotes;
  - identifying the speaker;
  - assigning a certain mood to each clause;
  - implementing voice acting.

The collection of training data consisted of a corpus of children's stories by Hans Christian Andersen (available at http://hca.gilead.org.il).

2 The structure of our model

This is the main structures used in our model. Other classes are:

- Main – opening files, handling databases, reading input

- Parser – tokenizing and parsing
- EmoFilter – training and classification of clauses and their moods

Text the main structure, contains all information about clauses, words, quotes and speakers.

> `all_words` – a vector of *Word* objects comprising the text
> `quotes_tree` – the root node of the *Quote* tree structure
> `all_clauses` – a vector of *Clauses*
> `speakers` – a vector of *Speaker* objects

Quote a tree of quotes with an artificial root representing the whole text

> `begin_idx` – index to this quote's begin marker *Word*
> `end_idx` – index to this quote's end marker *Word*
> `inner` – a vector of nested *Quotes*
> `speaker` – the *Speaker* of this quote

Speaker represents a speaker that's actually used in a quote, contains voicing parameters. Each speaker is associated with one *PossibleSpeaker* object, which provides more information such as gender, number and name.

> `baseline, rate, volume` – parameters for pitch, rate and volume
> `name` – a link to the original *PossibleSpeaker*

PossibleSpeaker represents a noun type, a speaker that might be realized in a quote. When references are analyzed, a single name may have more than one *Reference* object but only one real referent –one *PossibleSpeaker* which contains information about the single prospective speaker which all references of the same name are associated with. A *PossibleSpeaker* may of course not be realized in a real *Speaker*

> `name` – the string name
> `refs` – all the *References* , all the instantiations of this referent
> `score` – the accumulated score of all the `refs`

Reference represents an instantiation of *PossibleSpeaker* , in other words, a reference for a single noun type *Word*

> `speaker` – the *PossibleSpeaker* associated with this reference
> `idx` – the index of the *Word* which this *Reference* represents
> `score` – the score, analyzed according to the `type`
> `type` – represents attributes of this reference relating to agreement and context

Anaphor a subclass of *Reference* , used for anaphoric nouns. Also contains (other than inherited fields):

> `ananame` – the original string name of this anaphor
> `resolved` – link of the *Reference* this anaphor is resolved to

Clause represents a single clause

> `begin_idx` – index to this clause's first word *Word*
> `end_idx` – index to this clause's last word *Word*
> `mood` – the result of the mood analysis for this *Clause*s

Word holds the string tokens (words, morphemes or marks) and information about them

> `word` – the string *Word*
> `tag` – Part of Speech tag *Word*
> `type` – the type of the word, used for punctuation marks (e.g. begin quote, sentence marker etc.)
> `ref` – the *Reference* associated with this word
> `is_split` – whether this word is actually a morpheme (e.g. the morpheme 'isn' of the word 'isn't')

EmoFilter Emofilter trains by receiving the classification of the user and stores all the words in the hash table accordingly to the class they belong to. In this way we know how many times it was in the test set per class. If we divide this through the total amount of words in the class we have the P(w—C). Then on classify we get all the relative counts for all the words for each class and multiply them and determine the highest value this one we return.

Moods •

- neutral
- happy
- sad
- angry cold
- angry hot

3 The groundwork

3.1 Parsing

In order to do a good job of speaking the text, text-to-speech (TTS) systems need some knowledge of the structure of the text to be spoken. Thus, the analysis of the input text is required in order to find out relevant aspects that help to improve the output.

In our application, once the input is read, parsing is performed in order to distinguish between normal words and marks. Punctuation marks are carefully analyzed, in order to identify beginnings and endings of quotations, sentence and paragraph markers, and other marks which are usually taken as clause markers. All of these different marks are crucial for different parts of the application.

Sentences Delimiting sentences is not a trivial task in general (e.g. abbreviations, initials etc.). In our domain, stories often contain sentences that extend over quotes, and quotes that extend over sentences, which makes it tricky to plan. We decided not to base our application on whole sentences, however, being that the POS-tagger was trained on the whole sentences, it was still desirable to present it with sentences. Therefore, a simple pattern matching is applied.

Quotations This is one of the most important parts of our work. Quotations on the web are nicely presented by special characters for begin and end quote. Not wanting to lose this valuable information, input is read using the standard extended encoding, *UTF-8*, with which many web pages are encoded. This special characters are read, converted to our system-internal meta-language, and used to delimit quotes and later construct the *Quote* structure.

Clauses Clauses are important for our mood module. This is a much simpler task, performed using standard pattern matching. Clauses are later organized into a *Clause* structure.

Once the text is parsed, a *Text* object is created, which contains a vector of *Word* objects, wherein each word is represented and meta-tags are represented within the word's *type*.

3.2 POS-tagging

In order to determine possible speakers, verbs, informative words and so on, our application depends on knowledge of the function of the words. A POS-tagger is a simple yet very successful way to achieve this. We decided to incorporate a Viterbi-type stochastic POS-tagger, that was written by Ori Garin in collaboration with Fivos Karalis as a midterm project for the 'Language and Speech Processing' course at UvA. This tagger was trained on the 'Wall Street Journal' tagged corpus and it achieves 96% precision.

The POS tagger requires some simple morphology which is already performed by pattern matching while parsing (e.g. he isn't → he isn 't). The *Text* object produces the input for the tagger using a function that respects these requirements. Then special sentence tags are inserted between sentences, and the parser is invoked per sentence. Its output is then parsed tag per tag and distributed to the *Word* objects in the text.

3.3 Quote Analysis

For our application, quote-annotated text segments were defined as quoted speech. Given the distinction between beginning and ending quotes, this task was realized in a recursive way:

```
new Quote(index)
    this.begin = index
    index++
    while (index < #words)
        if quote begins at index
            next = new Quote(index))
            add next to the list of inner quotes
        if quote ends at index
            this.end = index
            return
        index++
```

Our quote analysis doesn't assume that all the quotes will come out fine. Therefore when a begin or an end quote is encountered, the module checks if it is the right mark (e.g., a quote like: '. . . " is taken as incorrect). All the quotes that fail this analysis, as well as all quotes for which no end quote marker is found, are removed from the quote structure. This analysis has proven to be very successful, and in our chosen domain of stories it doesn't fail.

Of course, identification of speech quotes is not enough. Sophisticated techniques of text analysis are also needed to identify which speech portions should be read by which speakers. Zhang et al. (2003) report on the system, called ESPER, which takes as an input raw children's stories and marks up who speaks when. More generally, this research group explores how to automatically identify spoken text within a story and, by identifying the characters in the story, to assign each quote to a particular character. ESPER is a component of the StoryTeller project, a collaboration between Carnegie Mellon University and Oregon Graduate Institute, which focuses on speech synthesis for children's stories. One of their goals is to be able to automatically select all potential speakers and to single out an appropriate speaker voice for the synthesis of each quote. Festival provides much of the infrastructure within this project.

4 Speaker Analysis

4.1 The approach

Zhang et al. offer the following characteristics which might ease speaker's identification:

- the speaker is referenced in the same paragraph where the quoted-speech occurs;
- character names within the quotes are not the speaker even though some speakers refer to themselves in the 3rd person singular.

Thus, if there is a character name preceding the quoted speech within the same paragraph, assign it as the speaker of the quoted speech. Otherwise, use the named character following the quote. Here the problem arises when the distance between the character and the speech exceeds the paragraph scope. Another trouble is inability to find any speakers at all for a given piece of quoted speech.

In our case, the approach for finding out the speaker of a quote is based on some observed properties in children's stories. In order to be able to find the correct speaker with some reliability, we had to take into account the following:

- In stories with many characters, the name of the participants is often repeated to avoid confusion.
- In stories with one main character, a pronoun is used profusely and consistently after its introduction, so that a child would be able to remember the original reference after several sentences.
- Quite many quotations in the stories follow the same pattern:

  . . . ?" thought the emperor. ". . .

  . . . !" he exclaimed.

  . . . ," said the other cucumber. ". . .
- Children's stories have a complex syntax!

The fact that not every chunk of quoted speech is an independent unit is also very important because several neighboring pieces of quoted speech may require the same speaker. Thus, it is necessary to detect whether a piece of quoted speech is a new quote (a different speaker) or a continuation quote (the same speaker as the previous quote). Zhang et al. established features which help to classify the quoted speech as a new quote or a continuation quote. They are the following:

- the word string preceding the quote;
- the word string succeeding the quote;
- capitalization of the first word in the quote;
- whether the quote starts paragraph;
- whether the quote is the first quote in the paragraph
- the end-punctuation of the previous quote within the same paragraph;
- the punctuation of the word preceding the quote;
- the length (number of words) of the gap between the last quote and the current one (within the same paragraph).

They concluded that the most reliable predictor of quoted speech types is the capitalized feature. The punctuation of the word preceding the quote is also reliable although in a less amount.

Obviously, in selecting the potential speakers, proper names can be identified first. However, it's not sufficient. What we came across in H.C.Andersen stories is a considerable number of characters who are not named, but referred to by descriptions (e.g., the peasant's wife). In this case, additional linguistic information is needed to make the proper identification. For example, the POS tagging can help to extract non-proper names (definite noun phrases can be potential character names). Unfortunately, the precision can still be low due to the fact that irrelevant entities in the story might be thought to be character names. (Although it is better to overlook then to miss a character).

Our speaker analysis is based on reference analysis, a "speaking" verb database, anaphora resolution and speaker identification. Saliency is realized by a score which all submodules use and modify.

4.2 Reference Analysis

This part is based on POS tags and on the verb database. The words are scanned to find nouns and anaphora, each of which is scanned and scored according to its context. A higher score is given for:

- anaphoric NPs
- subject NPs
- NPs which appear right before a verb
- NPs which appear right next to a speaking verb
- proper names

A low score is given to:

- NPs which appear right after a preposition
- possessive NPs

Each anaphor is stored in an *Anaphor* object, the other nouns are stored each in a *Reference* object. Each *Reference* is linked to a *PossibleSpeaker* object which represents an entity in the story, so *Reference*s with the same name will link to the same *PossibleSpeaker* . After the analysis described above, each *PossibleSpeaker* will contain the combined score of all *Reference*s that link to it. In order to use saliency as a bias for anaphora and speaker resolution, the (combined) score and reference count of each *PossibleSpeaker* is distributed back to each of its *Reference*s to augment their score.

4.3 Anaphora Resolution

Our shallow approach implies that much information is missing, such as gender, sentence structure and some familiarity factors. Our aspiration is to complement this lack by topicality and saliency represented in the score.

The analysis starts by marking all the words which appear within a quoted speech. We do not consider them. The other words are scanned in order to find ones that are linked to an *Anaphor* object. For each anaphor the words are scanned backwards and possible *Reference*s are collected. The score of each such reference may be augmented due to distance and coreference. The following rules were implemented:

Gender

- male or female - the scan continues until a reference is found
- neutral - the scan goes only as far as one sentence back

Distance

- represented as a (negative) bias which is added to the score of possible *Reference*s for resolution
- a reference in the same clause will get a big negative bias, unless the anaphor is possessive or reflexive.
- when a sentence marker or a quote is reached, the bias is decreased
- when a clause marker is reached, the bias is very slightly decreased

Agreement

- a reference must not differ in gender and number from the anaphor
- when an anaphor is finally resolved, it contributes its gender to its reference, except for neutral gender anaphora which is a more complex and problematic case

Coreference

- the list of possible resolutions never contains two *Reference*s that are resolved to the same *PossibleSpeaker* . Instead, when such a second *Reference* is considered possible for this anaphor, it adds its score to the first *Reference*

After a list of possible resolutions is collected, along with their new scores, the list is sorted and the most highest scoring *Reference* is selected.

4.4 Speaker Identification

The process just described for anaphora resolution is somewhat simpler for quotations. Although speaker identification and anaphora resolution have some differences, the considerations to be taken in our score-based saliency approach are quite similar. The main distinction in children's stories, however, is that in quotes, the speaker is often mentioned after the quoted speech whereas cataphora is probably never used.

There are some helping factors in quoted speech, the most notable one is the case where quoted speech is followed by a continuation of a sentence (i.e. no capitalization or punctuation marks). The speaker identification module first assesses which directions to scan the quote:

```
if (token after quote is not a lowercase word)
    disable fwd-analysis
else
    if (a speaking verb appears in first clause after quote)
        disable bwd-analysis
```

Both analyses or one of them traverse a specified stretch of text, seeking for words which are linked to a *Reference* object[1]. These references are collected into a vector, then sorted and the highest scoring one is chosen as speaker.

- forward analysis is done only on the first clause after a quote, up to 2 references are collected
- backward analysis scans up to one sentence back. The scan is ended when a quotation or paragraph marker is encountered or when a sentence marker is encountered for the second time.

5 Prosody

After identifying speakers and their pieces of speech, it is essential to make speech sound natural and human like. The problem of modern TTS systems is the relative lack of interesting and natural variability. Prosody is the most difficult aspect of generating synthesized speech. It covers the pitch, speed, range and volume with which syllables, words, phrases, and sentences are spoken. A sentence can be separated into a number of intonation units. As a rule, they can be identified by looking for punctuation marks (such as commas), but generally these kinds of surface cues might be unavailable and then the intonation is related to the syntactic structure.

Prosody provides broad parameters for speech synthesizers. That's why proper understanding and control of prosody can greatly improve naturalness of speech.

[1]This module is run after anaphora resolution. Being that *Anaphor* is a subclass of *Reference* , reference is used to refer both to anaphoric and non anaphoric NPs

The pitch and the pitch range can be set as “high”, “medium”,“low” and “default”. The range depends on a particular language and the voice. For example, female and children voices are usually higher than male voices. As a rule, a baseline pitch for male voices is between 140Hz and 180Hz, while female voices lie between 150Hz and 300Hz. So there can be different “defaults” within a group of female voices. Small pitch range makes speech sound monotone and flat.

The rate sets the amount of words spoken per minute. The descriptive values for rate are “fast”, “slow”, “medium” and “default”. Again, rate may vary across different languages. In English, normal speaking rates lie between 150 to 200 words per minute. To emphasize new information, one might slow down the rate and increase the pitch.

Finally, the volume attribute. Volume can be set as “loud”, “medium”, “quite” and “default”. In numeric values volume can be from 0.0(silence) to 1.0(loud). We increased volume whenever someone sounded happy or angry.

All four prosodic attributes have numeric equivalents to descriptive values discussed above. These numeric values will differ according to the language, speech synthesizer, etc.

In our application, we used the following modifications for different voices:

Prosody	Male	Female	Neutral	Plural
pitch	-10% to +20%	+40% to +80%	-20% to +70%	-20% to +20%
rate	+5% to +15%	+5% to +20%	-5% to +40%	-15% to +15%
volume	+5% to +20%	+5% to +15%	+10% to +25%	+15% to +35%

Thus, female voices had a higher pitch and faster rate than male voices. We expect to use neutral voices for birds, stones or anything else which can not be characterized as a male or a female, for this reason, there is a wide range of choice within the neutral voice. Finally, for the voices in plural (‘they said’) we did not make to much deviation from the narrator voice.

5.1 Emotional Speech

So far text-to-speech (TTS) synthesizers typically sound ”bored”, with little to no intonation or expression. Usually words and sounds are just cut and pasted together which results in clicks and unevenness that makes it difficult for children to understand what is being said. Another troublesome issue is lack of expressiveness. Severe modifications are needed to obtain lively emotional speech. But what is emotional speech? Following Alan Black (2003) emotional speech can be split into four moods: neutral, happy, sad and angry (hot/cold anger). Various studies have shown that listeners can hardly make distinction between happy and angry moods because in real life situations, lexical issues and context are the biggest cues to the emotional state of the speaker. Additionally, style in which the information is delivered greatly influences human perception.

To correctly identify the mood in which a piece of text should be synthesized, total interpretation seems unavoidable. For example, sentences can be deemed sarcastic on the feature that they are clearly violating common knowledge ("Oh, sure and everybody knows how lenient the Spartans were!"). However, due to the lack of feasibility and for experimental purposes, we have decided to take a shallow approach. We classified emotional speech using as little world knowledge as possible. One of the big advantages of such a system is that it can be trained, which means that it is supposed to be scalable from user to user and from language to language.

5.2 'Naive Bayes'

In document classification or, more precisely, spam filtering, 'Naive Bayes'('NB') has been widely adopted as a useful approach. The main characteristic of 'NB' is that it assumes feature independence, it can not form more complicated formulas between the features (like OR or XOR). In the field of document classification, 'NB' would not discriminate between 'Pedro hits the donkey', 'the hits Pedro donkey' and 'the donkey hits Pedro' (while obviously the latter would signal a completely different message). Although some search engines do incorporate a notion of distance and placement, but certainly not on this level.

Following an online encyclopedia 'Wikipedia', even though classifiers of this design often make blatant simplification assumption (which are often needed to make it feasible to design or compute), they often perform much better in many complex real-world situations. Recently, some theoretical grounds have been carefully studied to explain the surprisingly good results.

What is it we classify? Very early in building the application we decided that only characters partaking in the story should express their emotions. While the narrating character should be generally more impartial. We also noticed that speaker turns can be either very short ('Oh') or extremely lengthly, and even though this is not a problem for the classifying method of choice that is length independent (though modifications exist), it is a problem for us, since we do not want entire aliena read out in the same mood. That is why we have chosen our notion of clause as the resolution for this problem.

A word for word approach would be pointless and not helpful. It would be not helpful since it would sound like a very neurotic voice, and pointless since it would negate the 'NB' assumption of independent features and would just classify one feature at a time. We think sentences would be too big as a resolution since each clause can have its own mood variance.

What did we change to meet our requirements? Traditionally, 'NB' is used to discern between two classes (spam or not-spam, relevant or irrelevant). However, we used Black's classification of moods, that is we modified 'NB' approach. The probability of belonging to a certain mood class (Ci) given a certain clause was derived in the following way

$$P(Ci|wj...wn) = P(wj|Ci) * .... * P(wn|Ci)$$

After finding the scores for all classes within one clause, we compared the scores and picked the argMax. Our second modification to the original 'NB' is that we considered only nouns, proper names, adjectives and adverbs. This was done in order to avoid large scores for functional words. It is interesting to note that e.g., personal pronouns, can be argued to carry out some information about the mood as well. For example, a repetition of the pronoun 'you' in the following example might indicate that 'mother' is angry: "You, Mike, how many times should I tell you, what you should do!!!".

6 Suggestions for Improvement

Due to lack of time, many ideas were not implemented, and many other problems not addressed at all. Below we present some ideas that came up and some phenomena as well as shortcomings of our model.

6.1 Nested Quoted Speech

Although we have been working with child-oriented stories, the structure of the text was quite complex. For example, nested quoted speech is a regular phenomenon in H.C.Andersen stories, when one of the characters is narrating a story of his/her own, with its own set of characters and quoted speech, basically creating a story within a story. One such example is the fairy tale ""The Will-o'-the-Wisp Is in the Town," Says the Moor-Woman":

> And this is what the Moor-woman told:
> "There was a great commotion yesterday out here in the marsh! There was a christening feast!. . .
> "Now, all this I know, and all this I told to the twelve little Will-o'-the-Wisps whom I had on my lap, and who seemed quite crazy with joy.
> "I told them that the safest and most convenient course was to give up the honor, and do nothing at all; but the little flames would not agree to this, and already fancied themselves clad in fiery yellow clothes, breathing flames from their throats.

> “ ‘Stay with us,’ said some of the older ones.
> “ ‘Carry on your sport with mortals,’ said the others.
> “ ‘The mortals are drying up our meadows; they’ve taken to draining. What will our successors do?’
> “ ‘We want to flame; we will flame-flame!’ cried the new-born Will-o’the-Wisps.
> “And thus the affair was settled.
> “And now a ball was given, a minute long; it could not well be shorter. The little elf-maidens whirled round three times with the rest, that they might not appear proud, but they preferred dancing with one another.
> “And now the sponsors’ gifts were presented, and presents were thrown them. These presents flew like pebbles across the sea-water. Each of the elf-maidens gave a little piece of her veil.
> “ ‘Take that,’ they said, ‘and then you’ll know the higher dance, the most difficult turns and twists-that is to say, if you should find them necessary. You’ll know the proper deportment, and then you can show yourself in the very pick of society.’
> “The night raven taught each of the young Will-o’-the-Wisps to say, ‘Goo-goo-good,’ and to say it in the right place; and that’s a great gift which brings its own reward. . . .

In this example it is very hard to identify the (sub)speakers. It is also unclear whether such speech should be spoken in the quoted speaker’s voice, or should be different from the main speaker. Finally, this example shows that in the cases of nested quoted speech, quoted speech is marked by single quotes which in regular cases identifies labels (such as “. . . the street was called ‘Brinton’ ”). Thus, in such cases there is no obvious distinction between quoted speech and quoted labels. However, it is necessary to differentiate these two distinct cases so that quoted labels would not mistakenly be assigned a random character’s voice, which might cause confusion for the story listener. In an attempt to solve this problem we said that short quotes which are “uttered” via the verb ‘call’ are usually a label. We added a small ad-hoc feature that checks if such a condition is satisfied (with a limit on the number of words in the quote). As far as we have tested it, we obtained a desired result. Nevertheless, a more general solution for testing whether a quote is really a label is in place. In the future, it would be also nice to be able to automatically differentiate such labels from quoted-speech and synthesize them using pre-defined voice (such as that of the narrator or the main character) (Zhang, J.Y. et al., 2003).

6.2 Further Research

Our analysis of references only considers single nouns. Although it scans each noun’s context, much information is still ignored. Our simple analysis is extendible to consider different references of the same type (e.g. the yard cock and the weather cock; a young man, an old man and another man), but we felt it was beyond the scope of our work.

Our anaphora resolution is only performed on words marked as pronouns by the POS tagger. We noticed many other words which are or may be anaphoric but have not had time to address them.

Our model doesn't perform analysis of other coreferential expressions and familiarity. There are many examples in which this is a non-trivial problem (e.g. in 'The Emperor's New Suit', 'swindlers' and 'weavers' are used interchangeably).

In our corpora of stories a lot of sounds were used, e.g., "peep" - the sound of a new born duck; "hubble-bubble-hish" - the sound of a boiling pot; or "pop" - the sound of a shot. In the future it would be nice to have a database with predetermined sounds.

It is also unclear how to deal with rhymes and songs which are frequently used in children literature. For example,

> "...and the little bird, who wished to be amiable, began to sing,-
>
> > "Chirp and twitter,
> > The dew-drops glitter,
> > In the hours of sunny spring,
> > I'll sing my best,
> > Till I go to rest,
> > With my head behind my wing."..."

The narrative style should be investigated in more depth. We tried to play with breaks and emphases, however, modern synthesizers do not have a sensitive enough interface.

Finally, the mood classifier does not take into account the context of the clause which would add more precision.

7 References

1. H.C.Andersen stories were taken from http://hca.gilead.org.il

2. Black, A. (2003). Unit Selection and Emotional Speech, Eurospeech 2003, Geneva, Switzerland.

3. Hand, DJ, and Yu, K. "Idiot's Bayes - not so stupid after all?" International Statistical Review. 2001. Vol 69 part 3, pages 385-399. ISSN 0306 7734.

4. Zhang, J., Black, A., Sproat, R. (2003). Identifying Speakers in Children's Stories for Speech Synthesis, Eurospeech 2003, Geneva, Switzerland.