A LANGUAGE RESOURCES GENERATION TOOLBOX FOR SPEECH SYNTHESIS

David Sündermann

Siemens Corporate Technology, Munich, Germany
Universitat Politècnica de Catalunya, Barcelona, Spain
University of Southern California, Los Angeles, USA
david@suendermann.com

ABSTRACT

When creating voices for concatenative speech synthesis, several hours of speech uttered by a professional speaker are recorded. In order to make sure that the synthesized speech produced by concatenating segments (diphones, triphones, or even larger segments) of the recorded utterances feature a high naturalness, it is important that most of the segments required in the synthesis are available in the recorded speech. This is done by carefully selecting the sentences that are to be read by the speakers to achieve a high coverage of the required segments. In general, these sentences are extracted from huge text corpora of several domains. Furthermore, a good prosodic coverage is not less crucial for performing high-quality speech synthesis. Consequently, prosodic characteristics as the stress level that depend on the sentence type or on the part-of-speech categories have to be taken into account during the selection process.
In this paper, a language resources generation toolbox is presented that is able to cope with this task. As an example, the toolbox is used for generating the UK-English text corpora for speech synthesis in the European project TC-Star.

1. INTRODUCTION

One of the objectives of the European project TC-Star (Technology and Corpora for Speech-to-Speech Translation) [1] is the generation of corpora for text-to-speech synthesis. According to the TC-Star speech synthesis language resources specification [2], these corpora are twofold: they consist of text corpora (C) on the one hand and speech corpora, i.e. recordings of professional speakers reading the text corpora, on the other hand.

The speech corpora will be used for performing unit selection-based speech synthesis whose quality strongly depends on the presence of the required units in the application phase. The optimal solution to determine these units is to take all texts that are to be synthesized in the application phase into account. However, in the scope of TC-Star, a variety of domains with unrestricted vocabulary is to be taken into account (e.g. parliamentary speeches, novel reading).
In the language resources specification of the related European project LC-Star (Lexica and Corpora for Speech-to-Speech Translation), similar domains (sports/games, news, finance, culture/entertainment, consumer information, and personal communications) were covered [3], and lists containing with the most frequent words (about 50,000) of all considered languages were generated. Hence, these lists can serve as a reference when determining the set of units that must be covered by the TC-Star text corpora.

To fulfill the coverage criteria defined in [2], the author took large text corpora consisting of several million tokens (running words) from the two main domains parliamentary speeches and novel reading and selected appropriate sentences to build C. To perform this selection, a number of text and language processing steps had to be carried out. For this reason, the author established the language resources generation toolbox presented in this paper. It consists of system-independent Perl scripts and contains tools for

- text preprocessing: a tokenizer, tools for text normalization and for removing stress or special characters,
- processing lexical resources (for phonetic and syntactic lexica): type conversion, merging of several lexica, extracting sub lexica according to certain criteria,
- respelling (e.g. changing a text's spelling from US to UK-English),
- statistics (of lexica and plain texts),
- grapheme-to-phoneme conversion,
- sentence selection (greedy algorithm-based),
- statistical part-of-speech tagging (including tools for training and evaluation),

This work has been partially funded by the European Union under the integrated project TC-Star - Technology and Corpora for Speech to Speech Translation - http://www.tc-star.org.

I would like to acknowledge the contributions of Ute Ziegenhain and Thomas Volk to this work.

- format conversion for part-of-speech resources and lexica.

In the following sections, it is shown how the toolbox is applied to the generation of the TC-Star speech synthesis text corpora for UK-English.

2. BASELINE LANGUAGE RESOURCES

As already mentioned in the introduction, for creating the text corpus C, several language resources are required as, in particular,

- a phonetic lexicon to transcribe plain texts (the *phonetic* coverage is to be optimized), cf. Section 2.1
- large text corpora of the target domains that are suitable for selecting appropriate sentences, cf. Section 2.2
- a manually tagged text corpus to train a part-of-speech tagger (the part-of-speech information is necessary for determining the *prosodic* coverage), cf. Section 2.3

In this section, the above listed language resources and some pre-processing steps are briefly described.

2.1. The Phonetic Lexicon

To obtain a phonetic lexicon that is sufficiently large so that grapheme-to-phoneme conversion becomes unnecessary for the coverage investigations, the author combined the contents of two UK-English lexica, the Celex [4] and the Unisyn [5]. In doing so, he assumed that the following normalizations do not essentially affect the coverage measures that form the fundament of the investigations presented in this paper:

- removing all special characters from the lexical tokens and converting all lexical tokens to lower case (by using `norm.pl`),
- removing all blanks from the lexical tokens (`replaceStr.pl`),
- after combining, keeping only one (preferably the most probable) transcription for each lexical token.

In practice, these steps are performed using the toolbox scripts as follows[1] (the symbol `"..."` denotes a continuation of the line)

```
cut -f2 celex > celex.f2
cut -f1 celex | norm.pl | replaceStr.pl | ...
paste.pl celex.f2 > celex.norm.noBlank
```

`unisyn.norm.noBlank` is generated accordingly. Now, the lexica are joined:

```
cat celex.norm.noBlank unisyn.norm.noBlank | ...
joinLex.pl > lex
```

Celex and Unisyn and, hence, `lex` have two stress levels (high stress indicated by `'"'` and low stress indicated by `"'"`). However, for several investigations as for instance the triphone coverage experiments described in Section 3, the stress is neglected, thus, here, the lexicon `lex.noStress` is introduced:

```
replaceStr.pl '["'"']" < lex > lex.noStress
```

As already mentioned in the introduction, a lexicon that follows the LC-Star convention is to be created to serve as a reference for the coverage experiments described in the following sections. For this purpose, a wordlist is provided that is a suitable estimate of the most frequent words from the target domains. The second column of this wordlist contains the respective words' frequencies counted in a very large multi-domain corpus as specified in [3]. Our sublexicon is the selection of all entries from `lex` that are contained in `wordlist` after a normalization according to the above steps:

```
cut -f1 wordlist | norm.pl | replaceStr.pl | ...
lookUp.pl lex > sublex
```

Furthermore, by applying the above procedure, a sublexicon without stress information is derived: `sublex.noStress`. To create a count table with all words in the lexica, the counts provided by the wordlist are copied, and all words not contained in this list are assumed to be singletons (the minimum count of the wordlist is 2):

```
cut -f1 lex | paste.pl 1 | replace.pl wordlist > ...
lex.count
```

The number of different triphones in `lex` is determined by

- transforming the phonetic transcription to triphones,
- extracting the statistics (`lex.tri.stat`),
- and counting the number of its entries:

```
cut -f2 lex | phone2tri.pl | stat.pl > lex.tri.stat
wc -l lex.tri.stat
```

To create triphone statistics that take the above derived word counts into consideration, the script `stat.pl` can be called

[1] All lexica in this paper are expected to be in table format, i.e., the token is separated from its transcription by a `"\t"` character. The phonemes are separated by blanks (`" "`). For lexical format conversion, see the documentation of the functions `lex2tab.pl`, `tab2lex.pl`, and `moveStress.pl`.

	celex	unisyn	lex	wordlist	sublex
tokens	167,510	118,374	141,567	51,581	42,915
words	97,058	116,739	141,567	51,581	42,915
|words|	92,078	112,743	141,567	49,775	42,915
triphones			21,077		12,969
singletons			3,229		3,253
triphones'			37,507		20,070
singletons'			8,135		6,489

Table 1. Lexicon statistics. Tokens are word-transcription pairs. The number of tokens is greater than or equal to that of the words since one word can have several transcriptions. |words| is the number of words after normalization. triphones' and singletons' take two stress levels into account.

with the second column of `lex.count` as argument and results in the real-weighted statistics `lex.tri.stat.real` The number of singletons can be extracted by selecting the words with a count of at the most 1:

```
cut -f2 lex.tri.stat | select.pl 0 1 | wc -l
```

For the statistics of the mentioned lexica, have a look at Table 1.

2.2. Large Text Corpora

According to the specification of the TC-Star speech synthesis corpora, in particular two domains are to be covered: parliamentary speeches and read novels. Consequently, for the large text corpora, the following texts were chosen:

- The UK-English part of the European parliament plenary sessions (EPPS) corpus [6], version February 24, 2005, in the following referred to as `epps`.
- A collection of 31 English public domain novels from the 19th and 20th century, in the following referred to as `novel`.

To generate the corpus statistics, the toolbox also provides suitable functions. For instance, in the following, the number of different triphones in `novel` is determined by

- first normalizing the corpus (`novel.norm`),
- generating the phonetic transcription using a look-up in `lex`,
- transforming the phoneme sequence to triphones,
- extracting the statistics (`novel.norm.tri.stat`),
- and, finally, counting the number of its entries:

```
norm.pl < novel > novel.norm
cat novel.norm | lookUp.pl lex ' _ ' | ...
phone2tri.pl | stat.pl > novel.norm.tri.stat
wc -l novel.norm.tri.stat
```

	epps	novel
tokens	30,366,389	1,776,202
words	91,435	17,012
singletons	33,632 (36.8 %)	2,910 (17.1 %)
OOV tokens	613,440 (2.0 %)	24,626 (1.4 %)
OOV words	46,141 (50.5 %)	11,581 (68,1 %)
triphones	13,660	11,708
triphones*	9,385 (96.6 %)	8,633 (88.9 %)
	9,553 (98.3 %)	

Table 2. Corpus statistics.

To determine the number of corpus tokens not contained in the lexicon (out-of-vocabulary, OOV), the above applied lexicon look-up function `lookUp.pl` is used in mode `1`, i.e., the words that could not be successfully transcribed are output:

```
cat novel.norm | lookUp.pl lex '' 1 | wc -l
```

According to the specification [2], for the triphone coverage experiments discussed in Section 3, only non-singleton triphones of `sublex` are considered (the investigations of this section neglect the stress levels, i.e., `sublex` must be replaced by `sublex.noStress`, and so on):

```
cut -f2 sublex.tri.stat | select.pl 2 | ...
index2line.pl sublex.tri.stat | cut -f1 > ...
sublex.tri.noSingle
```

In Section 3.2, the respective real-weighted statistics are required:

```
cat sublex.tri.noSingle | lookUp.pl ...
lex.tri.stat.real|paste.pl sublex.tri.noSingle|...
reverse.pl '\t' > sublex.tri.noSingle.stat.real
```

To estimate the maximum triphone coverage that can be achieved by selecting sentences from the large corpora, the number of different triphones (in Table 2 referred to as *triphone**) that are shared by the lexicon (without singletons) and the respective corpus is determined:

```
cut -f1 novel.norm.tri.stat | ...
lookUp.pl sublex.tri.stat.noSingle | wc -w
```

2.3. The Part-of-Speech Corpus

For the experiments on prosodic coverage described in Section 4, information about a sentence's content and function words is required. For this purpose, the toolbox con-

	wsj
tokens	1,061,772
words	46,806
singletons	21,552
tags	45

Table 3. Part-of-speech tagging corpus statistics.

tains the statistical part-of-speech tagger *synther* presented in [7]. It requires a corpus to train its statistics. As the spelling convention (UK or US English) does not seem to play an important role when performing part-of-speech tagging, the Wall Street Journal corpus `wsj` is used for training. This corpus follows the Penn Treebank tagging guidelines [8] and served as training material of several well-studied part-of-speech taggers as the taggers of Eric Brill [9] and Adwait Ratnaparkhi [10]. The corpus statistics are displayed in Table 3. To train the tagging statistics, the script `trainSynther.pl` is used (the training corpus must be in the format proposed by Ratnaparkhi [10], i.e., one sentence per line, word and corresponding tag are connected by underscores `"_"`):

```
trainSynther.pl < wsj > wsj.synther
```

3. PHONETIC COVERAGE EXPERIMENTS

In the TC-Star language resources specifications, the corpus to be generated is broken down as shown in Table 4. The order in that the respective corpora are processed when optimizing the phonetic coverage results from the following considerations:

- The corpora C1.1 and C3.1 are already given: The first is a parallel text from the parliamentary domain translated from Spanish, the second are hand-written phrases that are used very frequently and should not be missing when building a speech synthesis text corpus.

- The corpus C3.3 is to maximize the frequencies of rare phonemes, it is generated independently of the triphone coverage.

- The corpus C2 is to consist of short sentences from the novel corpus to support the synthesis of spontaneous speech (here, the sentence prosody – that should be vivider in short sentences – plays an important role). The sentences are to contain between 10 and 15 tokens:

  ```
  cat novel.norm | lineLen.pl | select.pl ...
  10 15 | index2line.pl novel > novel.10-15
  ```

corpus	domain	tokens	order
C1.1	parallel parliamentary speeches	9,000	0
C1.2	parliamentary speeches	36,000	3
C2	novels	27,000	2
C3.1	frequent phrases	8,000	0
C3.2	triphone coverage sentences	8,000	4
C3.3	mimic sentences	2,000	1

Table 4. Breakdown of the corpora to be generated.

  However, this heavily reduces the amount of available text: After selecting only short sentences, the number of tokens decreases to 40,397. Consequently, the novels should be processed before the EPPS, as the latter provides essentially more and phonetically richer material.

- The corpus C3.2 is designed to achieve the required coverage even though the remaining corpora's coverage is lower. The approach is to take words from the lexicon that contain still missing triphones and generate sentences with them. This can be done by searching respective sentences in huge databases as the Internet or by manually writing.

3.1. Maximizing Rare Phonemes

As aforementioned, the corpus C3.3 that is to consist of 2000 tokens aims at maximizing the frequency of rare phonemes. Since the term *rare* is not clear, the TC-Star specification defines that a minimum count of 10 per phoneme has to be achieved. The phoneme set is based on the computer-readable phonetic alphabet SAMPA [11] and consists of 47 phonemes.
As the fundamental corpus, all sentences with a length of 10 are selected from `epps` as described above and result in `corpus.10` and `corpus.norm.10`, respectively. The latter contains 28,626 sentences. After transcribing these sentences, a greedy algorithm [12] is applied (`greedy.pl` in mode `1`) that iteratively selects the sentence with the greatest count of the rarest phoneme of the already selected sentences (again, all applied lexica in this section are the `noStress` variants):

```
cat corpus.norm.10 | lookUp.pl lex | greedy.pl ...
1 | index2line.pl corpus.10 | head -200 > C3.3
```

Here, the first 200 lines (i.e. the first 2,000 tokens, cf. TC-Star specification) are the corpus C3.3.
The rarest phoneme (`"e@"`) occurs 48 times.

corpora	triphones	coverage
C1.1 + C3.1 + C3.3	3,813	39.2 %
+ C2.part1	5,504	56.6 %
+ C1.2.part1	8,188	84.3 %
+ C3.2.keyword	9,018	92.8 %
final	9,089	**93.5** %

Table 5. Triphone coverage.

3.2. Maximizing the Triphone Coverage

According to the TC-Star specification, the whole corpus to be created (C1.1, ..., C3.3) has to cover at least 90 % and if possible 95 % of the non-singleton triphones of `sublex.noStress`. From Table 1, the latter can be derived: $n_{ns} = 9{,}716$. As already argued in Section 3, the coverage achieved by greedily selecting sentences from the large corpora will be lower than the aforementioned numbers since, by means of the corpus C3.2, missing triphones are considered using manually generated sentences. C3.2 is to consist of 8,000 tokens. When specifying a sentence length of 10 tokens, one obtains at least $n_{C3.2} = 800$ additional triphones that can be deducted from the above numbers.
For instance, when the goal is to achieve a reasonable coverage as $\hat{c} = 92.5\,\%$ (the average between minimum and recommended coverage), the greedily generated part's coverage is only

$$c' = \hat{c} - 100\,\% \cdot \frac{n_{C3.2}}{n_{ns}} = 84.3\,\%\,,$$

i.e., about $n' = 8{,}188$ different non-singleton triphones of `sublex.noStress` must be contained in the remaining corpora.
In Table 5, the respective triphone coverages of the corpus generation steps described in the following are shown. The already existing corpora C1.1, C3.1, and C3.3 result in less than half of the target coverage c'. The triphone statistics of these corpora based on the full lexicon `lex` is `C-3.3.tri.stat`.
For the triphone coverage optimization, this time the script `greedy.pl` is called in mode `0`, i.e., it iteratively selects that sentence which contains the most symbols not yet covered by the already selected sentences. In doing so, it takes additional weights into account:

- The sentence lengths of the input corpus should be considered, as most uncovered triphones occur in very long sentences that would be preferred. More reasonable is to select that sentence, whose ratio between the number of not yet covered triphones and its length is maximum (`weight1`).
- To achieve a maximally natural triphone coverage, not only the fact that they occur but also their occurrence frequency is to be taken into account. This is done by providing the counts of all triphones of the input corpus as their respective weights. Furthermore, this allows for excluding triphones that are covered by the already given corpora by assigning zero to their weight (`weight2`).

According to Section 3, the sentence selection starts with corpus `novel.10-15`:

```
lineLen.pl < novel.norm.10-15 > C2.part1.weight1

cut -f1 C-3.3.tri.stat|paste.pl 0>C-3.3.tri.stat.0
cut -f1 lex.tri.stat.real | paste.pl 0 | ...
replace.pl sublex.tri.noSingle.stat.real | ...
replace.pl C-3.3.tri.stat.0 > C-3.3.tri.stat.real
lookUp.pl C-3.3.tri.stat.real < ...
novel.norm.10-15.tri > C2.part1.weight2

greedy.pl 0 C2.part1.weight1 C2.part1.weight2 < ...
novel.norm.10-15.tri > C2.part1.greedy
```

The output of the script `greedy.pl` contains 5 columns:

- the indices of the selected sentence,
- the above defined ratio,
- the latter's numerator
- and denominator,
- and the number of non-zero-weighted triphones covered by the respective sentence which are not yet covered by prior sentences.

Having a look at `C2.part1.greedy`, one observes after a number of lines that the second column becomes zero. This is a sign that all triphones of the input corpus are already covered. The following procedure extracts all contributing sentences:

```
cut -f2 C2.part1.greedy | select.pl 0.1 | ...
index2line.pl C2.part1.greedy | index2line.pl ...
novel.10-15 > C2.part1
```

`C2.part1` consists of 9,961 tokens, hence, 17,039 tokens remain from the C2 corpus for the prosodic coverage experiments discussed in Section 4.
Now, the same procedure is applied to the parliamentary speech corpus `epps`. As the TC-Star specification defines a minimum sentence length of 25 tokens for the C1.2 corpus and the whole `epps` is too large to be processed by the `greedy.pl` script in a reasonable amount of time, the sentence lengths are limited to between 25 and 30 (`epps.25-30` consists of 4,561,055 tokens). The greedy algorithm produces the output file `C1.2.part1.greedy`.

Taking the number of triphones that are covered by the already generated corpora and the target n' into account, one obtains the number of triphones that are to be covered by the currently generated part of the corpus C1.2: $n_{C1.2} = 2{,}648$. As aforementioned, the fifth column of the greedy algorithm's output file `C1.2.part1.greedy` gives each selected sentence's contribution to the coverage and now is to serve for determining the size of `C1.2.part1`:

```
cut -f5 C1.2.part1.greedy | sum.pl 1 | select.pl ...
0 2648 | index2line.pl C1.2.part1.greedy | ...
index2line.pl epps.25-30 > C1.2.part1
```

Finally, the 800 keywords that are needed for the C3.2 corpus are determined. This is done using the same procedure as before, however, as input, `lex.tri`, the triphone transcription of all entries of `lex`, is utilized. Due to the greedily extraction of the keywords, the 800 words deliver 930 not yet covered triphones, thus, the coverage rises to 92.8 % which is 0.3 % more than expected, cf. Table 5.

4. PROSODIC COVERAGE EXPERIMENTS

The experiments described in this section deal with some word and sentence characteristics that have influence on the word or sentence prosody. They concern the stress information of particular words in a sentence, their position within the sentence and the sentence type. As in Section 3, all these phenomena are expressed by an optimization of triphones with extended attributes:

- Now, three stress levels are to be distinguished. However, the stress only plays an important role in conjunction with the sentence's first content word or the last ones in each phrase. For the definition of sentence and phrase, see Section 4.1.
- The position of triphones in a sentences is described as either *prepausal* or *non-prepausal*. According to [13], prepausal triphones are those between the last stressed triphone and the end of the phrase. Furthermore, when maximizing the coverage of prepausal triphones, the phrase type is taken into account. Three types were distinguished: *interrogative*, *exclamatory* and *other* phrases.

4.1. Including Stress Information

As mentioned above, the stress level is only distinct with particular content words in the sentence or phrase, respectively. For determining a text's content words and sentence or phrase breaks, the already mentioned part-of-speech tagger is applied. As this tagger follows the Penn Treebank part-of-speech specification [8], its input has to consist of well-defined tokens rather than plain text. For instance, contractions have to be resolved (`won't` $\longrightarrow$ `wo n't`), and punctuations have to be separated from the neighbored words. For this purpose, the text has to be preprocessed by a tokenizer.

Now, the tagger's output following the Ratnaparkhi format (cf. Section 2.3) is analyzed by the script `ratna2stress.pl` that replaces the part-of-speech tags by the following stress tags (in doing so, the algorithm expects one sentence per line – e.g., use the tool `sentenceEndRecognition.pl` – and interprets the tags `"''"`, `"("`, `")"`, `","`, `"."`, `":"`, and `"``"` as phrase break delimiters):

- `"0"` for non-content words,
- `"1"` for content words,
- `"2"` for phrase break delimiters,
- `"3"` for content words that are likely to be stressed.

For the time being, only the stressed content words which are output by default when `ratna2stress.pl` is called without parameters is needed (`corpus` stands for an arbitrary input corpus):

```
cat corpus | tokenizer.pl | synther.pl ...
wsj.synther > corpus.ratna
cat ratna2stress.pl < corpus.ratna > corpus.stress
```

For determining the current triphone coverage including stress, one extracts the stressed words of the already generated corpora `C-1.2.part1.stress` (`C3.2.keyword` is excluded since the keywords' position inside the sentences is not known yet), computes the triphone statistics using the lexicon *with stress* information, and compares this set with the list of non-singleton stressed triphones as described in Section 2.2. The result is displayed in Table 6.

Now, for both, the novels and the EPPS corpus, the stressed words are extracted and greedy algorithms for optimizing the triphone coverage including stress are executed. This is done as discussed in Section 3.2 and results in the partial corpora `C2.part2` and `C1.2.part2`. As above, the size of the former is defined based on the fact that after a certain number of sentences, no additional contribution to the coverage can be achieved (`C2.part2` contains 3,432 tokens).

corpora	triphones	coverage
C1.1 + C1.2.part1 + C2.part1		
+ C3.1 + C3.3	7,383	54.4 %
+ C2.part2	7,763	57.2 %
+ C1.2.part2	8,666	63.8 %
final	8,791	64.7 %

Table 6. Triphone coverage including stress.

corpus	phrases	interrogative	exclamatory
C1.1	839	12 (1.4 %)	5 (0.6 %)
C1.2	3,848	39 (1.0 %)	4 (0.1 %)
C2.part1 + C2.part2	2,780	234 (8.4 %)	142 (5.1 %)
C2.part3	2,934	755 (**25.7%**)	563 (**19.2 %**)
C3.1	1,788	96 (5.4 %)	35 (2.0 %)
C3.3	256	12 (4.7 %)	0 (0.0 %)

Table 7. Interrogative and exclamatory phrases.

corpus	tokens	
	specified	effective
C1.1	9,000	8,942 (–0.7 %)
C1.2	36,000	36,134 (+0.4 %)
C2	27,000	27,030 (+0.1 %)
C3.1	8,000	8,052 (+0.7 %)
C3.3	2,000	2,010 (+0.5 %)

Table 8. The corpora's final token numbers.

`C1.2.part2` was to fill the remaining part of corpus C1.2 (7,014 tokens).

4.2. Sentence Position and Type

This section deals with the last task described above: the coverage optimization of triphones regarding their position in the sentences and the sentence type. Here, the script `ratna2stress.pl` is called in mode `1` (cf. Section 4.1) followed by `stress2pos.pl` that extracts prepausal and non-prepausal phrases (mode `0`: non-prepausal; mode `1`: prepausal). The output of this script contains three columns:

- the phrases,
- the indices of the sentences the phrases belong to,
- and the phrase break delimiter.

```
cat corpus.ratna | ratna2stress.pl 1 | ...
stress2pos.pl 1 > corpus.pos
```

When counting the respective tokens, it turns out that only 18.7 % of them are prepausal. This means that a coverage optimization with respect to the prepausal triphones is more important than that with respect to the non-prepausal since the latter should be reasonably covered anyway. Furthermore, the sentence type generally becomes manifest in the prepausal part rather than in the non-prepausal. Consequently, in this study, only the prepausal triphones were investigated. They are derived from the prepausal phrases by means of the script `prepausal.pl` after applying the text normalization and the transformation to triphones (using the phonetic lexicon *including stress*):

```
prepausal.pl < corpus.pos.f1.norm.tri > ...
corpus.prepausal
```

It turns out that some of the phrases become empty due to the normalization, the lexicon look-up, and the phoneme-to-triphone conversion. These phrases are now removed as described for the sentence selection in Section 3 resulting in files indicated by the suffix `"1-"`. Furthermore, the stress information is removed (since in this investigation it does not play a role), and the triphones are merged with the corresponding phrase break delimiters, hence, every triphone can virtually occur in three types: interrogative, exclamatory, and as neither of them. Finally, all phrases that belong to the same sentence are joined using the aforementioned sentence indices:

```
cut -f2 corpus.pos.1- > corpus.index
cut -f3 corpus.pos.1- | replaceStr.pl '[^?!]' > ...
corpus.type
cat corpus.prepausal.1-.noStress | paste.pl ...
corpus.type '' ' ' | paste.pl corpus.type '' | ...
paste.pl corpus.index | joinLex.pl 2 | ...
reverse.pl '\t' | sort.pl -n > corpus.postype
```

For the optimizations described in this chapter, only the remaining part of the corpus C2 (novels) is available. The investigations on the conventional triphone coverage and that including stress information have shown, that using the subcorpus `novel.10-15` as input of the greedy algorithm does not provide sufficient phonetic variety: After a certain number of sentences, the coverage did not further improve. In order to overcome this effect, the input corpus was extended to include all sentences of a length between 5 and 30 words, i.e. 215,161 tokens.
Now a greedy algorithm according to Section 3.2 is performed taking the contributions of the already existing corpora into account and yielding to the corpus `C2.part3`. It contains a great number of interrogative and exclamatory sentences as they were hardly covered by the other corpora, cf. Table 7.

5. VALIDATION

After generating the corpora, finally, the fulfillment of the underlying specifications was verified. In particular, the following items were examined:

- All corpora have the same format as in the source corpora.
- The corpora do not contain multiple sentences.

- The number of tokens (after normalization) corresponds to those defined in the TC-Star specification, cf. Table 8.
- None of the C3.2 keywords are contained in the other corpora.
- The sentence lengths agree with those defined in the TC-Star specification.
- The phonetic coverages agree with those defined in the TC-Star specification, cf. Section 3.1 and Tables 5 and 6 (*final* entries).

6. REFERENCES

[1] H. Höge, "Project Proposal TC-STAR - Make Speech to Speech Translation Real," in *Proc. of the LREC'02*, Las Palmas, Spain, 2002.

[2] A. Bonafonte, H. Höge, H. S. Tropf, A. Moreno, H. v. d. Heuvel, D. Sündermann, U. Ziegenhain, J. Pérez, and I. Kiss, "TC-Star: Specifications of Language Resources for Speech Synthesis," Tech. Rep., 2005.

[3] U. Ziegenhain, "LC-Star: Specification of Corpora and Word Lists in 12 Languages," Tech. Rep., 2004.

[4] R. H. Baayen, R. Piepenbrock, and H. v. Rijn, *The CELEX Lexical Database*, LDC, Philadelphia, USA, 1993.

[5] S. Fitt, "Documentation and User Guide to Unisyn Lexicon and Post-Lexical Rules," Tech. Rep., University of Edinburgh, Edinburgh, UK, 2005.

[6] C. Gollan, M. Bisani, S. Kanthak, R. Schlüter, and H. Ney, "Cross Domain Automatic Transcription on the TC-Star EPPS Corpus," in *Proc. of the ICASSP'05*, Philadelphia, USA, 2005.

[7] D. Sündermann and H. Ney, "synther - a New M-Gram POS Tagger," in *Proc. of the NLPKE'03*, Beijing, China, 2003.

[8] B. Santorini, "Part-of-Speech Tagging Guidelines for the Penn Treebank Project," Tech. Rep., University of Pennsylvania, Philadelphia, USA, 1990.

[9] E. Brill, "A Simple Rule-Based Part of Speech Tagger," in *Proc. of the ANLP'92*, Trento, Italy, 1992.

[10] A. Ratnaparkhi, "A Maximum Entropy Model for Part-of-Speech Tagging," in *Proc. of the EMNLP'96*, Philadelphia, USA, 1996.

[11] D. Gibbon, R. Moore, and R. Winski, *Handbook of Standards and Resources for Spoken Language Systems*, Mouton de Gruyter, Berlin, Germany and New York, USA, 1997.

[12] J. P. H. van Santen and A. L. Buchsbaum, "Methods for Optimal Text Selection," in *Proc. of the Eurospeech'97*, Rhodes, Greece, 1997.

[13] A. Bonafonte, H. Höge, I. Kiss, A. Moreno, D. Sündermann, U. Ziegenhain, J. Adell, P. D. Agüero, H. Duxans, D. Erro, J. Nurminen, J. Pérez, G. Strecha, M. Umbert, and X. S. Wang, "TC-STAR: TTS Progress Report," Tech. Rep., 2005.