TR-IT-0288

Recording New Databases For CHATR

Ekaterina Saenko Nick Campbell

December 1998

ABSTRACT
Producing new speech databases for CHATR, a multi-speaker, multilingual speech synthesis system, is an important task which in the past had been accomplished by having a speaker read an arbitrary text. This paper discusses ways to facilitate the recording process, in particular, the selection of the input text based on a set of phonetic units and a phonotactic model of the target language, and outlines the design of an interactive recording and labeling tool.

Recording New Databases For CHATR

Table of Contents

Introduction

CHATR is a multi-speaker speech synthesizer which works by concatenating segments selected from a pre-recorded speech database to form a new utterance. Thus, to introduce a new speaker, or language, to the synthesizer, a new speech database must be created.

Currently, in order to record a new database, speakers either read a short novel of their choice, or read a special set of sentences that guarantee maximum coverage of the sounds occurring in the language. Such sets for English are the *200 sentences* and *TIMIT* sentences. However, these sentences are usually not connected at all and are difficult to read for the speaker. The result is that the speaker's voice sounds bored and has an unnatural intonation. Also, because the text aims at the maximum entropy of phonetic information, its contents are too "dense" to provide sufficient variety in prosody of the more frequent sound sequences. Both these factors are detrimental to the synthesis.

If we do not use a special collection of sentences, we must rely on quantity of recorded data to provide a good coverage of the sounds of the language and of their various contexts. Of course, quantity does not necessarily mean quality. Ideally, we would like to be able to construct a text that is easy to read with sentences that are somewhat related in meaning, while at the same time ensures a balanced coverage of acoustic information. Furthermore, besides automatically constructing a text, it would be desirable to automate the entire process of recording speech, labeling the waveforms, and adjsting the remaining text to reflect the actual phonemes and prosody in the database.

In this paper we will attempt to first define what it means for a phonetic corpus to be balanced, using probabilistic language modelling, then study the balance in several existing databases, focusing on acoustic balance and comparing several ways to construct models based on different unit sets. Then we will suggest several methods for automatic text selection and describe step-by-step the general design of a recording and labeling tool.

Table of Contents Next

Definition of Database Balance

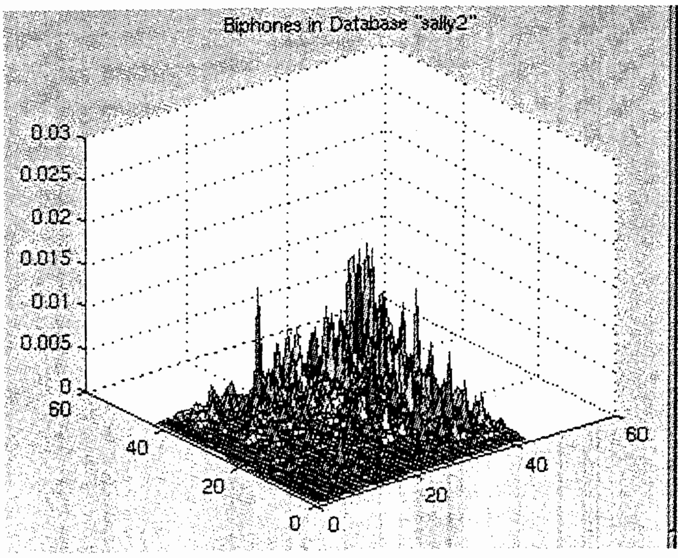
The Maximum Entropy Approach

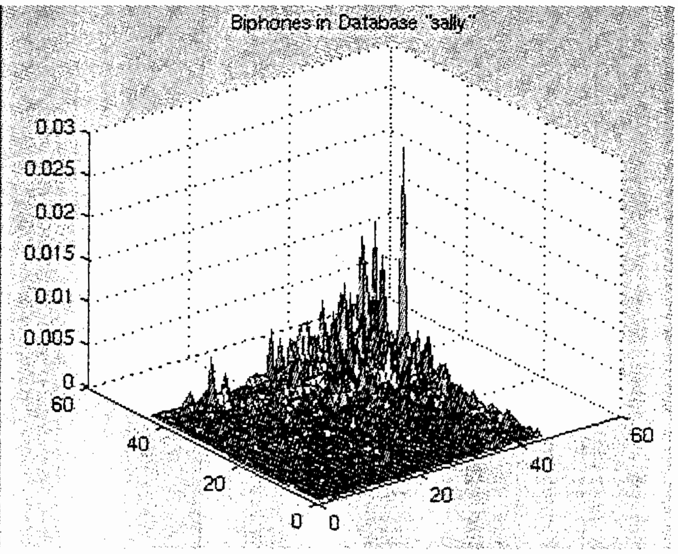
Database balance can be defined in several ways. In the past, there were attempts to acheive a balance of the phonetic content of the database. The goal was to maximize the number of various biphones recorded from reading of the minimum required text. The result was a text, consisting of several hundred isolated sentences, that was supposed to provide a balanced basis for recording. Some examples from the *TIMIT* set are

- *Ambidextrous pickpockets accomplish more.*
- *The willowy woman wore a muskrat coat.*
- *Cyclical programs will never compile.*

There are several problems with this approach to balancing a CHATR database. First of all, as evident from the examples, the sentences are not at all related in meaning and often resemble tongue-twisters in their difficulty of pronounciation. Experiments using the recorded sentences for CHATR databases showed that this lack of logic in the text translates to the speaker's voice sounding bored and monotonous.

Secondly, a more important flaw is that, in spite of being selected to maximize phonetic entropy, the resulting database is not better balanced than a regular CHATR database in terms of biphone content. For example, compare the two databases "sally" and "sally2", the first one of which is based on the *200 sentences* and the second is based on spontaneous speech.


Biphones in Database "sally2"


Biphones in Database "sally"

The plots above show biphone distributions in the two databases. Both are approximately the same in recording time. However, if we compare the ratio of unique biphone types per total number of biphones in the database, the non-balanced, spontaneous speech database "sally2" performs better than the balanced-set database "sally." The data are summarized in the table.

Type of Database	Size of Database (total num. of biphones)	Number of Biphone Types	Biphone Types Per Total Biphones
balanced sentences	8789	1172	0.1333
spontaneous speech	4995	757	0.1516

Another problem is that the balanced sentences are not geared toward any type of discourse, or, rather, they constitute their own kind of discourse, with tight, concise statements and oversimplified grammatical structure. If used for synthesis of speech in another context, such as spontaneous speech or newspaper articles, they may not have inadequate coverage of necessary units. For example, the most common biphone in the "200 sentence" database is *dh_@*, while in the "spontaneous" database it is not as frequent as, for example, the second most frequent biphone *ih_ng*. On the other hand, in "sally", biphone *ih_ng* is only one-fifth as common as *dh_@*. This shows that, depending on the type of discourse (spontaneous speech may have more gerunds like "going", "doing") the definition of phonetic balance changes. Thus, it would be useful instead of having a static text to be able to automatically construct a balanced text to suit a particular context, or many different contexts at once.

Finally, the balanced sentences are concerned only with phonetic balance, aiming at complete coverage of 2-phone sequences. But CHATR is a concatenative synthesizer that relies on the proper phoneme plus the required prosody to reside in the database, rather than on signal processing to modify pitch and duration. Thus it appears that balancing only acoustics is not the complete solution, and that prosodic balance should also be sought after.

Maximum Likelihood Approach

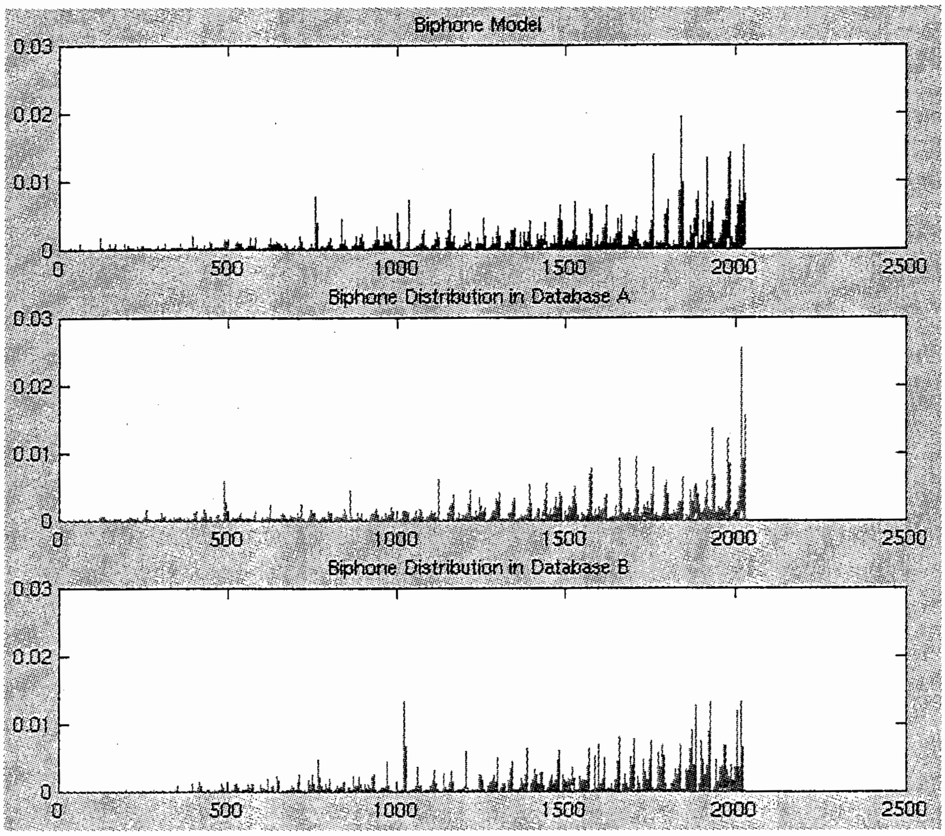
The new way of measuring balance in a database does not involve maximizing the variability of phonetic information, or phonetic entropy. Instead, the definition of balance is based on a statistical model of the phonetic stream. Such a model is trained on the corpus of text translated to phonemes with predicted prosody. Since the model reflects the particular context of speech and type of discourse, it is possible to adjust the definition of balance for synthesis in different contexts, from the entire language to just the credit card and phone number digits.

In order to compute the balance of a database, we need a reference model, or the "universe" model. Such a model is built from Maximum Likelihood estimates of model parameters and predicts the probability of each event (phonetic unit) in the given universe.

The graph below shows a historgam view of the biphone model, M, plotting the probabilities of each unit in the biphone unit set. Also shown are histograms of biphone units in two databases, A and B. If we define the balance in a database as the cosine of the two vectors corresponding to the "universal" distribution and the database distribution, then in our example

BALANCE(A,M) = 0.3594

BALANCE(B,M) = 0.1209



Using similarity to the universal model as a measure of balance, the distribution of units in databse A turns out better-balanced than database B. Scores computed using this measure range between 0 and 1. A higher score means that the database is made up of similar units to ones making up the universe.

-->

Acoustic Balancing of Input Text

The work involved in designing a synthesis database can be broken down into three stages. The first is deciding what units to use, the second is finding a statistical model that best describes the distribution of these units in the language, and the third is producing a database based on the model.

The main objective of this work is designing an inventory of speech segments that will provide a sufficient basis for future speech synthesis, while remaining within the storage requirements. In order to get a grasp on what needs to be in the inventory, we must do some form of phonotactic modelling on the target language to be able to predict what speech is likely to be synthesised from the database. Since the synthesis is done by concatenating segments from the inventory, we are specifically interested in predicting which segments will be used in future synthesis and with what probability.

Thus, the first stage, selection of the unit set, is a necessary step in describing the database. In the case of CHATR, whether or not these units will actually be the ones concatenated to form an utterance is difficult to assert. The units are chosen dynamically and there is no way at present to force the selection algorithm to use a fixed set of units. The range of units used is fixed only by virtue of that sequence of phonemes existing or not existing in the database. However, it is essential to define fixed units in order to be able to describe the composition and balance of the inventory. It is also a pre-requisite to training a probabilistic phonotactic model of the target language. Therefore, we will base our database design on a well-defined phonetic unit set and assume that this approach will benefit the UDB selection algorithm.

The average length of a segment (defined to be a continuous waveform taken directly from the database with no concatenations) used by CHATR's UDB is between 2 and 3 phonemic units. We will study and compare three different ways to construct a unit set:

1. using biphone sequences
2. using triphone sequences
3. using multiphones sequences (variable length up to N phonemes)

It would also be interesting to evaluate the performance of syllabic units, which are a type of variable length units. However, the bi-, tri- and multiphones in our study will be derived automatically, while it takes a linguist to manually segment training data at syllabic boundaries.

To compare the above three unit sets theoretically and experimentally, we determine what units occur naturally in the language. Since a massive training corpus necessary to approximate the target language is only available for English, that will be the language of choice in this study. The total number of naturally occuring units, as well as the relative frequency distribution of units and the number of units most commonly occuring in speech for each unit set will be analysed to determine the trade-offs between the sets. Furthermore, we try to predict how the use of a particular set of units might affect the quality of synthesis. In order to do this, we will need an objective measure of synthesis quality, based on which to compare the unit sets' performance.

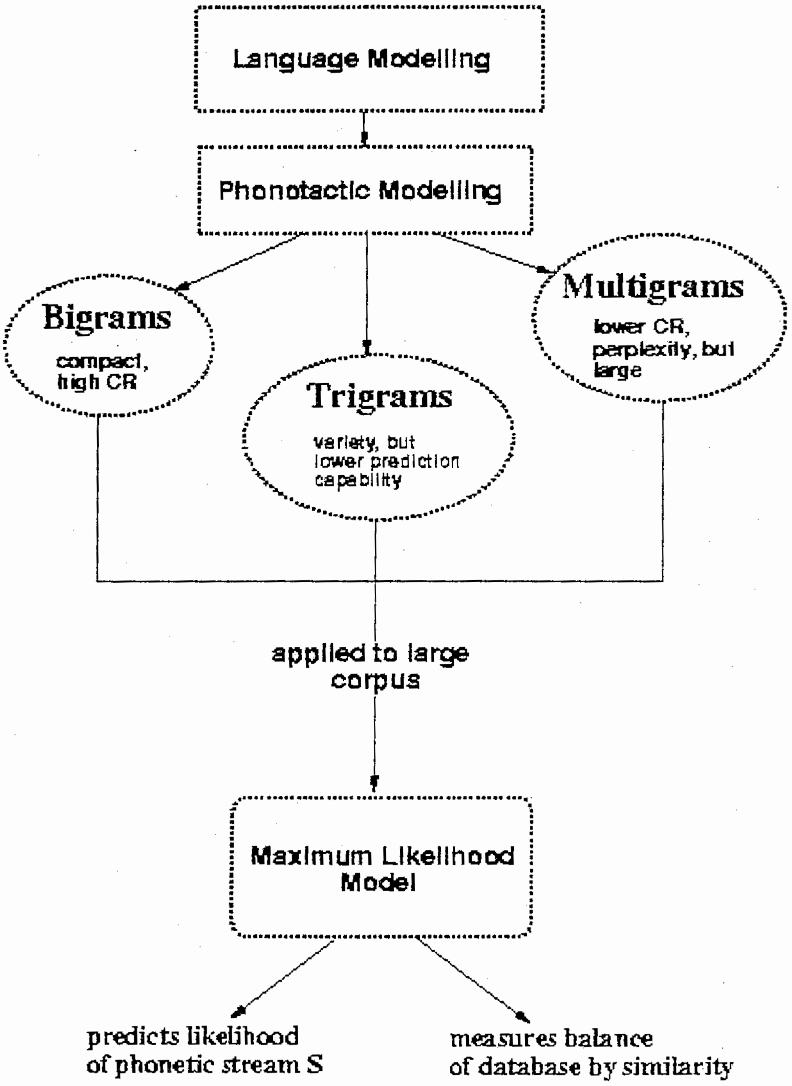

Language Modelling
Phonotactic Modelling
Bigrams
compact, high CR
Trigrams
variety, but lower prediction capability
Multigrams
lower CR, perplexity, but large
applied to large corpus
Maximum Likelihood Model
predicts likelihood of phonetic stream S
measures balance of database by similarity

Figure. Statistical Modelling of the Phoneme Stream

Once the unit set has been determined, the next step is to develop a stochastic model of speech segmented into those phonetic units. This can be done for English using phonotactic modelling tools on the available large training corpus. The speech selected for the CHATR database must reflect the statistical behaviour of units in the target language. Thus, in selecting the input text for recording, or in reducing a recorded corpus of speech to a smaller database, we aim to match the given model. Several algorithms for this *balancing* process will be examined.

We define the *balance* of a phonetic database by the similarity measure between the distribution of units in the database and that of the entire language. The language knowledge is approximated by the phonotactic model trained on the maximum sized training data for that language.

Modelling of Speech

N-gram Models

Human speech can be viewed as a stream of sounds emitted over a period of time, with meaning attached to particular combinations of basic sound particles, such as combinations of phonemes. This stream of phonemes is created by a probabilistic process which is not completelly random, as regularities in speech arise due to lexical, syntactical and other constrains.

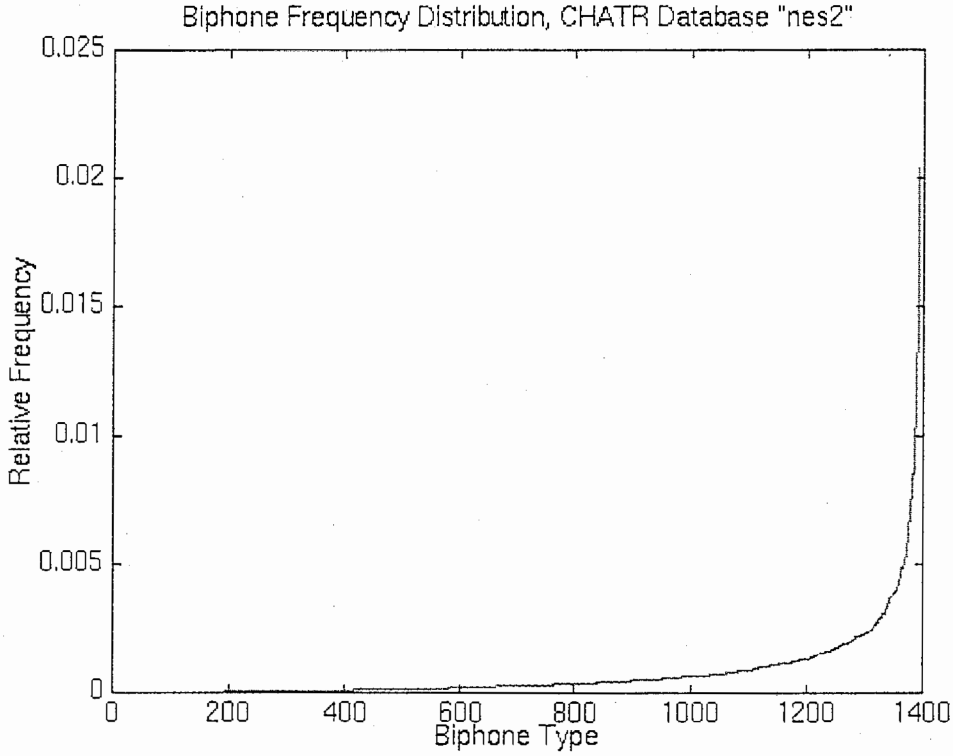
Concatenative speech synthesis works by breaking down the phonetic stream S into smaller segments and rearranging them to produce a new stream, S', which has different syntactical and semantical properties than S, but the same voice quality unique to that particular speaker. This method of speech creation takes advantage of the regularities of the phonetic stream, where the same patterns occur in various contexts, carrying different meaning. For example, English phoneme sequences "f aa s t f uu d" and "l aa s t w ii k" share a common subsequence "aa s t" that performs different lexical and grammatical functions in each sequence, yet may sound identical to a human ear.

Of course, speech is a far more complex phenomenon than can be described by a probabilistic process producing combinations of phonetic elements taken from a fixed set. Humans use several different sources of information to decode the message embedded in the acoustic signal. [SG] Variations in the spectrum, fundamental frequency and segmental duration all contribute to the complex structure of a signal. This is precisely why synthesis by concatenation cannot be accomplished by merely recording the basic phonemes of the target language: such a scant vocabulary will not be sufficient to reproduce the prosodic variety of all streams possible in the language. In fact, no fixed set of phonetic units will ever suffice, as human speech is infinitely variable. However, as experiments have shown, larger unit sets perform well enough if not for any imaginable utterance, then at least for the majority of use cases. Thus, our goal is to determine a set of units with maximal variability, while keeping database size in consideration.

There are several ways to model the stochastic process underlying the creation of a phonetic stream. Statistical language modelling by N-grams has long been used in speech recognition, and can be applied to a variety of problems, including phoneme sequence modelling. The approach taken by the N-gram model is to hypothesise that the probability of an event (in our case, phoneme) depends on the N preceeding events, with a constant N over the entire history of events. In applying the the N-gram model to unit set generation, we consider a set generated from 2-grams (or biphones) and 3-grams (or triphones).

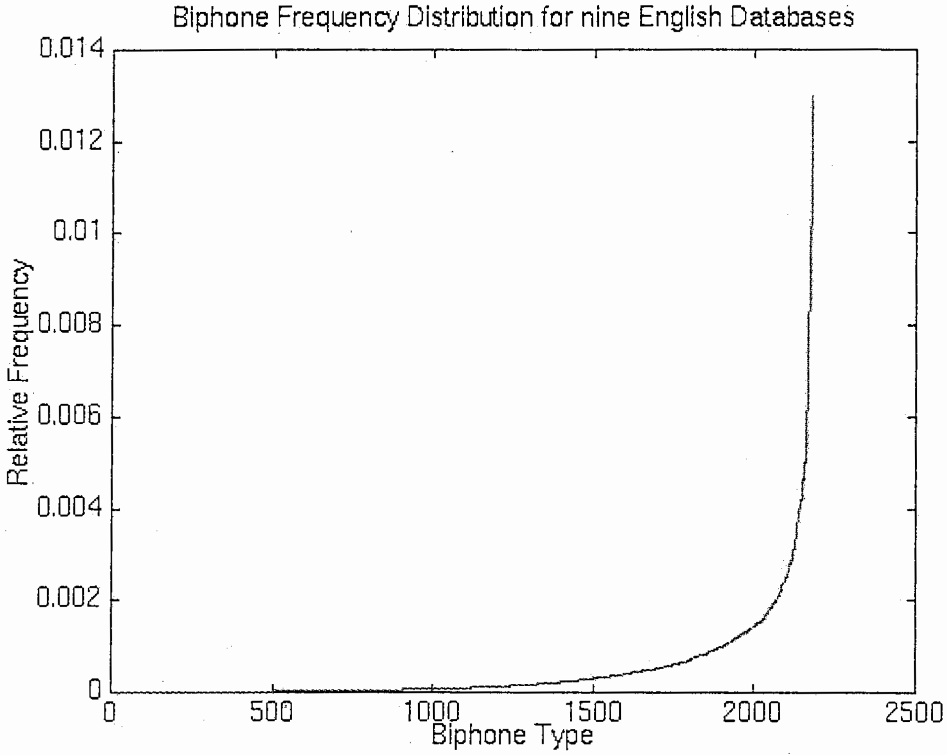
Biphone Frequency Analysis

First, we analyse the biphone units contained in actual recorded speech signals, such as CHATR speech databases. In the database, the speech waveform has been recorded and broken up into phonetic elements and labeled according to the phonetic transcription of the read text. As a first approach, we count the number of occurrences of each 2gram of phonemes. This analysis shows that the database for speaker "nes2" contains 35065 instances of biphones, of which 1391 are distinct. In the graph below, the 1391 biphones are plotted on the x-axis, and their relative frequency on the y-axis.



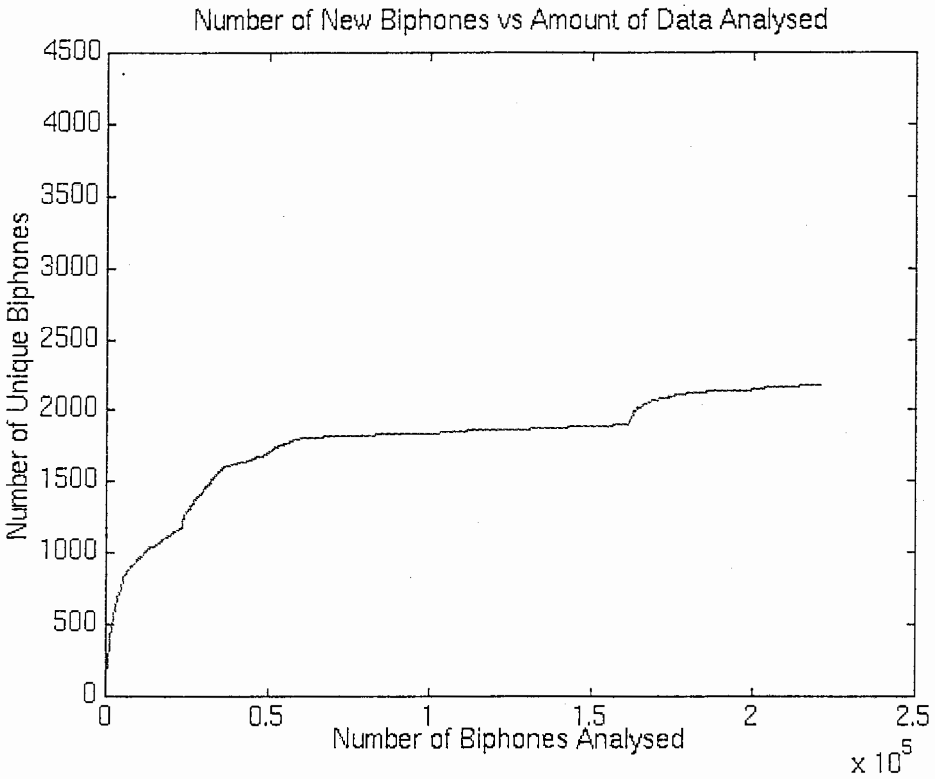
The frequency distribution plot shows that, by far, not all units are equally likely in read speech. The five most frequently occurring biphones, *'n_t'*, *'s_t'*, *'n_d'*, *'t_ax'* and *'ax_n'* have a much greater likelihood than the majority of occurring biphones.

Repeating the same analysis for a larger set of recorded data, we get the following results



As the graph shows, there are approximately 2200 unique biphones occurring in the available data. The question arises: how many biphones occur in speech in general? We know that, given a phoneme set of 56 unique phonemes (the *beep* set, not counting coughs, laughs, etc. and repeating symbols for breath and silence,) 3136 combinations are possible, which is not to say that all occur naturally. If we count all phonemes in the beep phoneset, then up to 4356 biphones are possible.

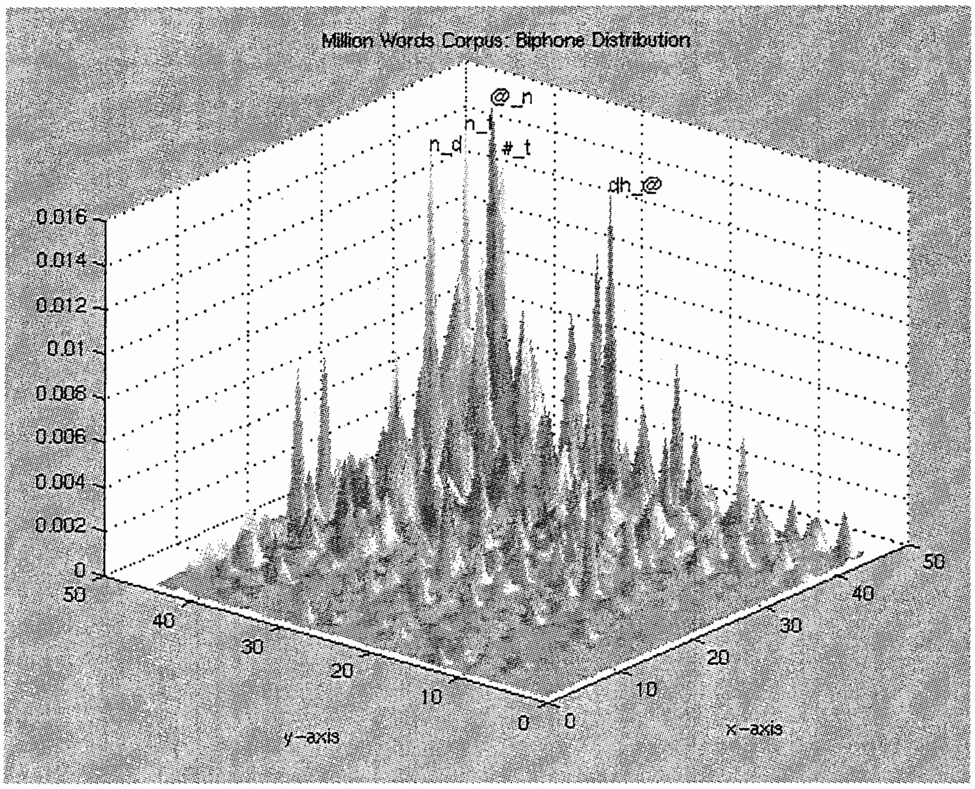
Perhaps if we look at the relationship between the number of phonemes analysed and the number of unique biphones extracted, we can estimate how complete our collection of unique biphones is. For example, if we find that we're discovering fewer and fewer new biphones with each new sentence we look at, then at some point we'll have the bulk of the most frequent ones.



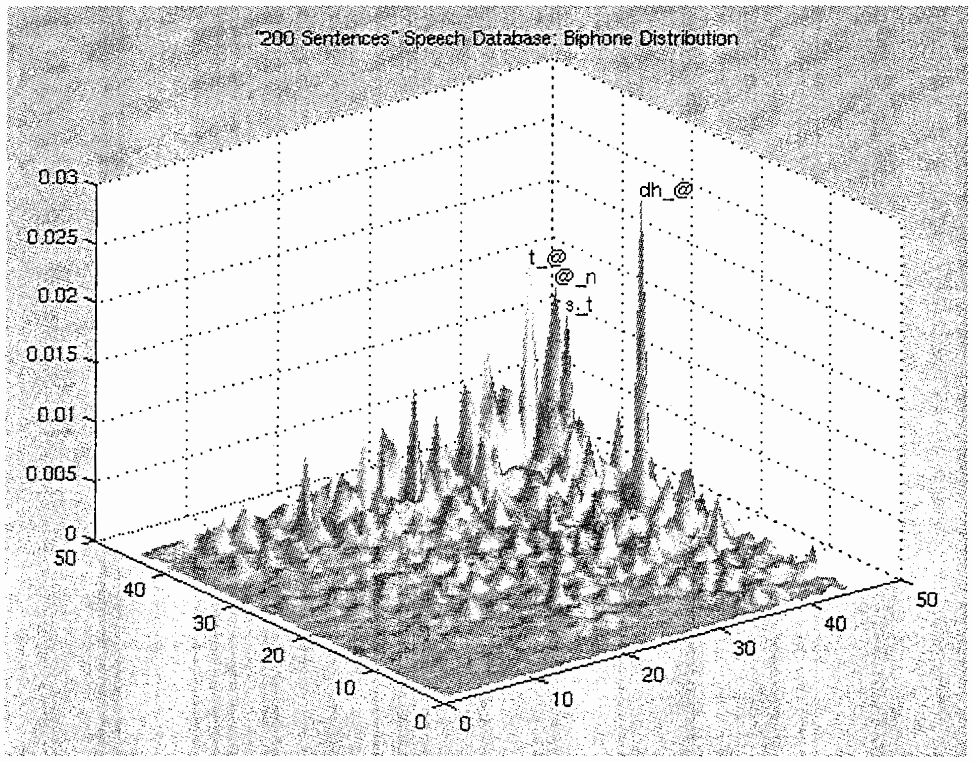
As the graph above shows, we would have to analyse a gigantic amount of data before we find all of the 4000+ biphones. The sudden rises in the otherwise smooth plot indicate points when we went from one database to the next. If the segment files were chosen at random, the graph would probably be much smoother. (Note: the phonesets used for these databases contain non-overlapping phoneme names)

Comparing "Balanced Sentences" With Spontaneous Speech

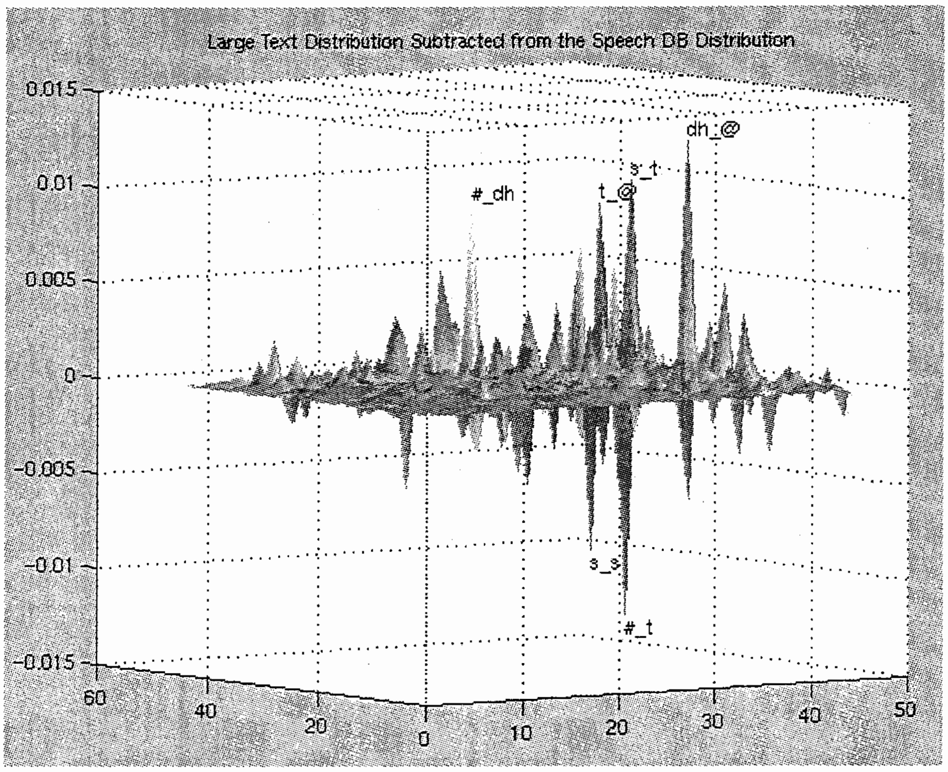
The following plot shows biphone frequencies for a very large text corpus. The axes are labelled with *mrpa* phonemes sorted from 1 to 45 in order of increasing frequency of occurrence in this text.



The figure below shows the count of each biphone in the *mrpa* phoneset for speaker "sally". The axes are labelled with *mrpa* phonemes sorted in the same order as above (1-45).



The following plot shows the difference of the two distributions.

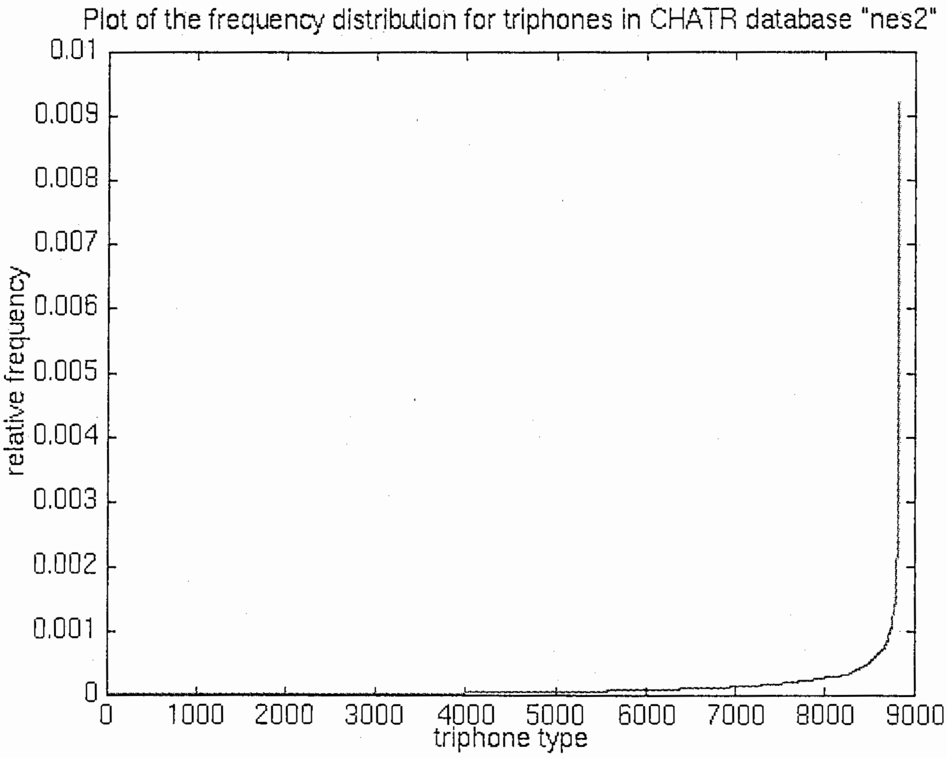


Comparing the distributions of biphone units in the language and in the database, the database distribution is unbalanced in the sense that some units, such as *dh_@*, are over-represented in the database, whereas other units, such as *#_t* (pause followed by phoneme "t") are under- represented. We want to balance the database by filling in these gaps, bringing the distribution closer to that of the language.

Triphone Frequency Analysis

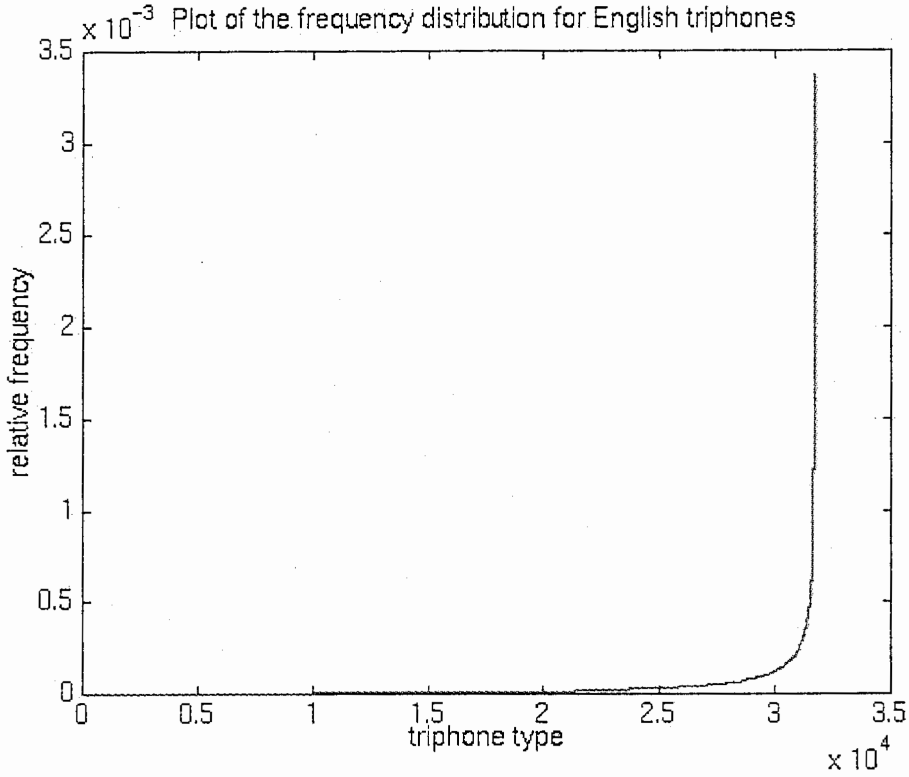
Analysing the frequency of 3-gram, or triphone, combinations in English databases produced the following results.

- The CHATR database for speaker "nes2" contains 35880 triphones, which can be divided into 8813 groups of triphones that differ acoustically from one another. Triphones within each group vary in prosodic quality, but we first analyse the frequency with which 3-phoneme combinations occur in the English language. The triphones are labelled from 1 to 35880 in order of increasing number of occurrences. The relative frequency of each triphone is computed by dividing the number of occurrences by the total number of triphones in the database.



Note that approximately 45% of all triphones in this database occur only once, or 0.000028 percent of the time.

- To illustrate what happens when the recorded speech data is very large, I analysed twelve English speakers' databases at once. The result was 31669 acoustically distinct groups out of a total of 270867 triphones. The following is the frequency distribution plot.



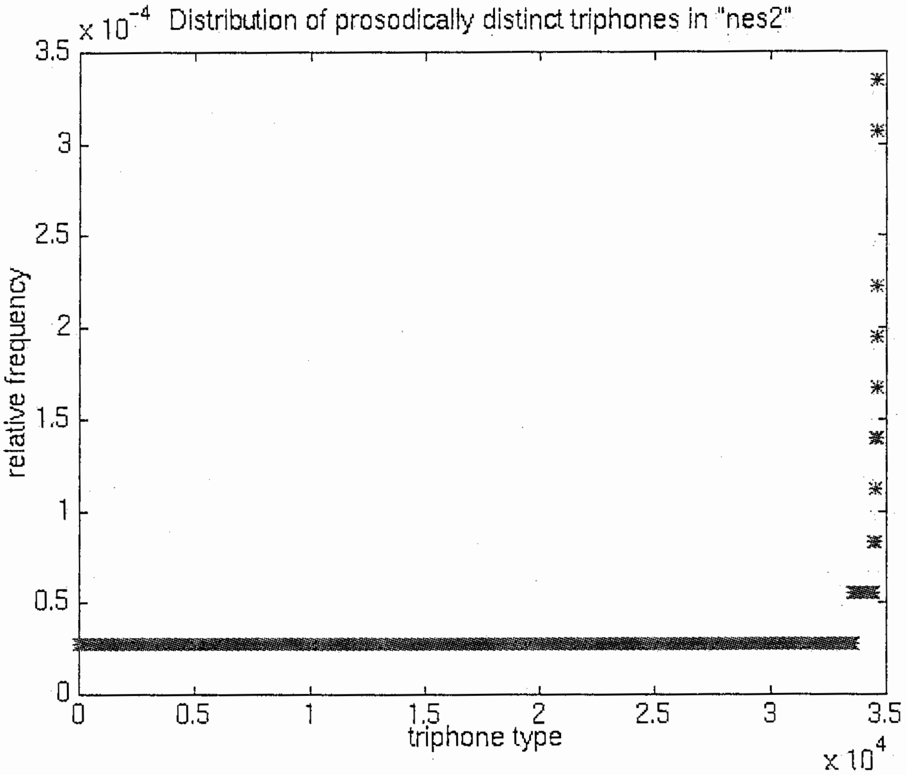
Since the number of different triphones is now four times larger than that in database "nes2", the highest relative frequency, 0.0034%, is lower than the highest frequency in the "nes2" database, 0.0092%.

- The above analysis shows that a large number of distinct three-phoneme combinations occur in the English language, but only a small number of triphones has a relatively high frequency distribution.

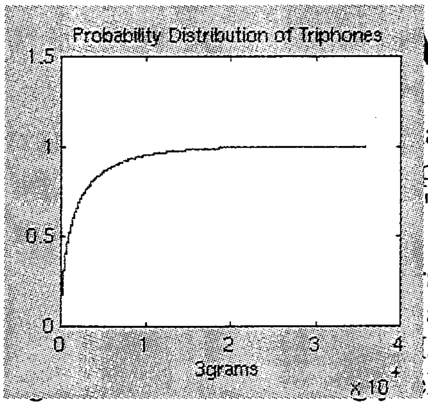
However, even data covering *all* possible triphones would not be sufficient for a balanced speech database. We need to look not only at the two surrounding phonemes, but at the prosodic characteristics of the central phoneme as well.

We choose to treat two instances of a triphone as identical if
- they have the same acoustic features (i.e. consist of the same sequence of phonemes) and
- their central phonemes have pitch and duration values that are within 0.1 units of each other.

Below is a graph illustrating the frequency distribution of such prosodically distinct triphones in the "nes2" database.



It's obvious that most prosodically distinct triphones occur very infrequently, in fact, among the 35880 triphones in the "nes2" speaker database, 34576 are unique! It is interesting to note that the combination of phonemes "ax", "n" and "d" is among the most frequent ones.



Multi-gram Models

ar, but is assumes that the statistical dependencies between symbols are
peech, some variable-length sequences of phonemes are equally likely,
', "o_v_dh_@" all have approximately the same likelihood.

s a string of symbols as the concatenation of independent
applied to speech synthesis unit generation, it produces multiphone
aximum Likelihood. The multigram model extracts variable-length
lated to the morphological structure of speech. For example, consider
two sentences parsed by the Langmodel tool using a bi-multigram model:

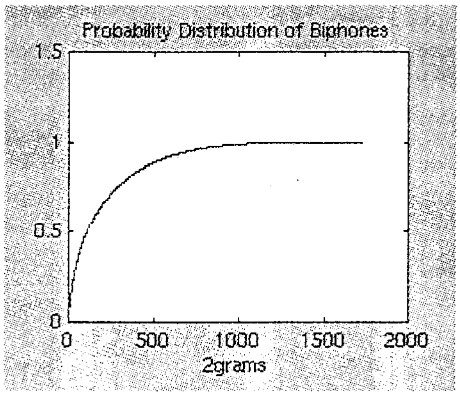
- The smell of freshly ground coffee never fails to entice me into the shop.

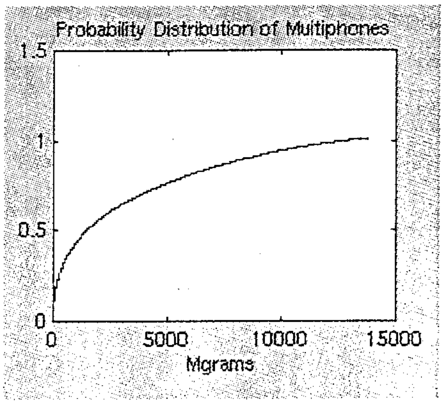
-
  ```
  pau_dh_@_s  m_e_l  o_v_f_r e_sh l_ii
  g_r_au_n_d k_o_f_ii pau_n_e_v_@ f_ei l_z
  t_@ e_n_t ai_s m ii_pau_i_n t_@_dh_@ sh_o_p
  ```

- The chill wind caused them to shiver violently.

-
  ```
  pau_dh_@ ch_i_l w_i_n_d k_oo_z_d  dh_e_m
  t_@_sh i_v_@ v_ai l_@_n_t l_ii
  ```

The table below compares 2-gram, 3-gram and 5-multigram unit sets.





Average length of multiphones (based on 13744) is N = 3.6978

Type of Ngram Unit	Total no. of units	No. of units that accounted for 50% of all instances	No. of units that accounted for 90% of all instances	Storage requirements (total no. of phonemes)
Biphone (N=2)	1,729	106	528	1,056
Triphone (N=3)	35,862	871	6,451	19,353
Multiphone (1<=N<=5)	13,744	1,443	8,460	29,924

Experiments have shown that a sentence can be synthesized from multiphones with twice less concatenations than from biphones [2]. As can be seen from the previous table, the storage requirements for a multiphone set are approximately 1.5 of the size of the triphone set. Overall, multiphones scale well compared to triphones, and they provide the longer but necessary sequences that tend to be re-used in speech often and therefore distorted in pronounciation.

Automatic Input Text Selection

Given any text and a set of units, we can compute its balance relative to the universal model. The computed value can be used to compare several texts in order to select the best one.

However, in many cases it is hard to find a suitable text, such as a novel, that can be used to record the database (for example, if the database is for synthesising phone numbers), or we would like to generate a smaller subset of a larger text. In order to ensure that the selected subset is comfortable to read for the speaker, we use their text of choice as a source text to reduce. For example, it may be easier to read several hundred of paragraphs from "War and Peace" selected to match the unit make-up of phone numbers than to read rows and rows of digits.

There are two ways to select optimal input text:

- start with an empty set and accumulate text items
- start with a full set of text items and reduce it

The next section describes a greedy algorithm to handle the first case.

Text Generation By Accumulation

Problem Statement.

Given N = {1,..,n}, a set of phonetic text transcription items, consider a finite collection of subsets {S1, S2, ..., Sm}. The feasible subsets are represented by inclusion or exclusion of items such that they satisfy certain conditions, in this case, that the total number of phonetic units contained in the set is limited by *U_max*. A variation of the problem limits the total number of items in the set Si.

Each subset has an objective function value f(Si). The objective function is defined so as minimising it would produce the desired property of the set. For example, we can define the function as the balance score of the set relative to a model M:

f(Si) = BALANCE(Si, **M**)

Alternatively, let

f(Si) = || **M** - D(Si) ||,

or the L2 norm of the difference vector between the given unit distribution **M**, and the unit distribution in the set Si. The problem is to minimize f(Si).

Approaches.

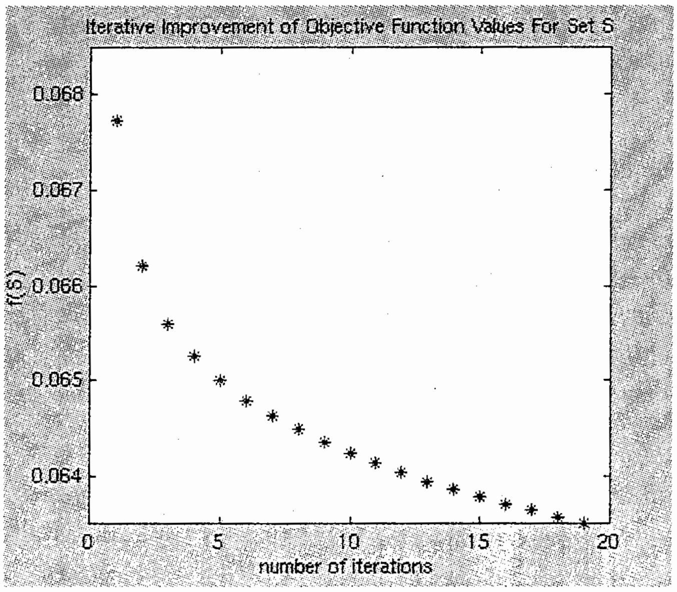
This is a *combinatorial optimization* problem. Some examples of this class of problems are the knapsack, the travelling salesman and shortest path problems. Various algorithms can be applied to solve CO problems, such as *greedy algorithms*, *dynamic programming*, *genetic algorithms*, etc.

Greedy Algorithm Approach

In mathematical programming, any (purposeful) search procedure to seek a solution to a combinatorial optimization, is called a *heuristic search.* Greedy algorithms are a specific class of heuristic search algorithms. A greedy algorithm begins with no elements and sequentially selects an element from the feasible set of remaining elements by myopic optimization. However, it is not guaranteed that this method will reach the optimum solution for a given problem.

The following is the output of a program that selects a subset S of text items (in this case, paragraphs) from a given file using a greedy strategy. The distance values are the f(Si) values at each iteration and the total number of units is also reported. The "-u" option specifies the length of the phonetic unit (in this case, a diphone.) The first paragraph selected is the one with the most units.

The graph shows distance values after each iteration; a total of 20 paragraphs from "Alice In Wonderland" are selected.



P118, P119, P119, P156, P165, P188, P202, P206, P209, P218, P240, P262, P262, P325, P346, P346, P346, P346, P408, P433

Total units: 3478

Drawbacks:

- does not reach the optimal solution
- improvement is slow for small domain (700 items)

To prove that the above algorithm does not reach an optimum solution, it is enough to consider a counterexample: if the optimum solution is the set S* and if there exists an item P_k **not** in S* such that f(P_k) < f(P_j) for all P_j in S*, then the algorithm will select P_k and therefore will not produce the optimum solution.

Backtracking

We can try to improve this heuristic method by backtracking at each step and considering the values of f(S_j - P_i), i != j. If the value decreases, we reject the item P_i from the set. The reasoning behind this is that we will have considered more combinations, at an increased computational cost, of course.

However, there is a very small improvement in using this backtracking technique: after selecting 18 items, the distance is 0.063629, compared to 0.063634 previously.

Another approach (which is guaranteed to produce the optimum solution) is to consider all possible subsets {S1, S2, ..., Sm} that satisfy the given constraints. This is not, however, feasible in most cases because of the computational effort involved.

Unit Subset Coverage

So far we have tried to match the entire unit distribution, which is inherently difficult to do when one is restricted to using "blocks" of a phonetic stream. If the number of blocks available is small, it may be impossible to find a combination that produces a similar distribution for **all** units. Instead, we may want to only guarantee coverage of the more important units.

An algorithm for selecting a subset of **S** that contains the units u in **U** is implemented as follows:

Compute a sparse matrix A of size nXm, where m is the number of units and A(i,j) is the count of the j-th unit in the i-th item os set S. Then, until all units have been included in the subset,

- Chose row k of A that has the largest intersection with set **U**
- If there is more than one k, choose one with minimum unit count
- update **U** by removing units appearing in the k-th row

When selecting a set of units **U**, we may want to select units making up the upper x percent of the cumulative distribution.

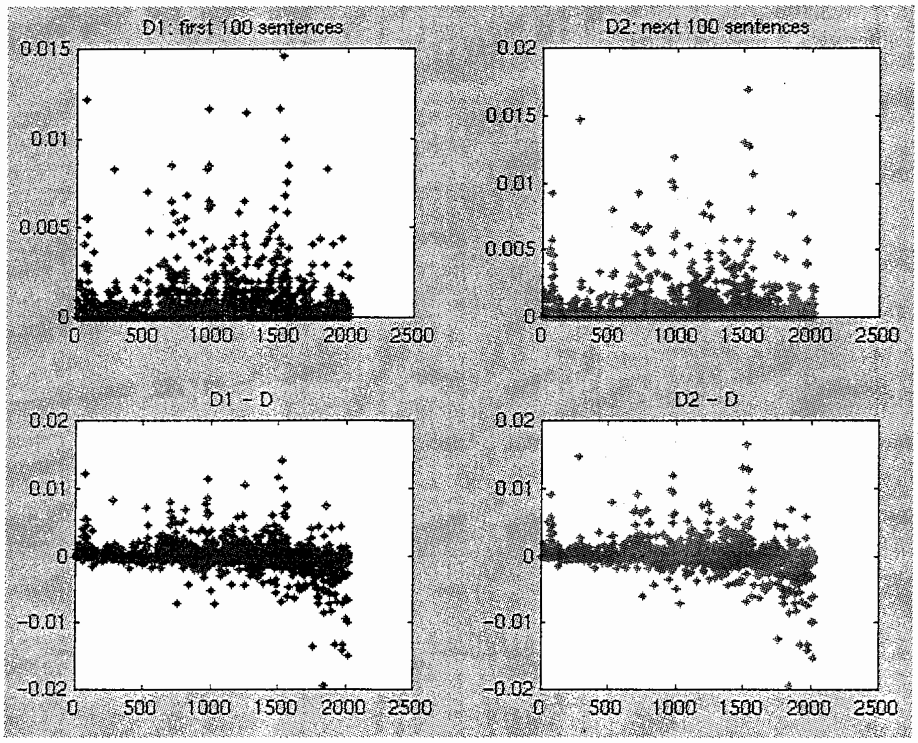
Text Generation By Reduction

If we have a text that is supposed to have many similar sentences, an effective technique for input text generation can be reducing the sentences based on their similarity scores.

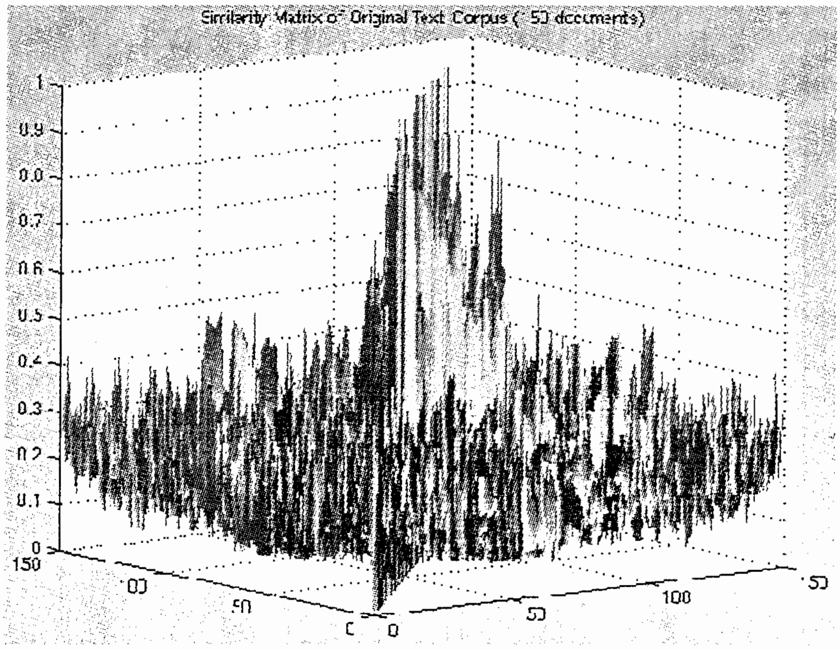
First, compute a weight vector for each sentence, or document. Each document then can be represented as a vector $d_i = (d_{i,1}, d_{i,2}, \cdots)$ where the individual elements are the frequency of each unit in the document with the model likelihood of this unit removed from it. Then the similarity score is

$$SIM_c(d_i, d_j) = \frac{\sum_{k=1}^{t}(d_{i,k} \cdot d_{j,k})}{\sqrt{\sum_{k=1}^{t} d_{i,k}^2}\sqrt{\sum_{k=1}^{t} d_{j,k}^2}}.$$

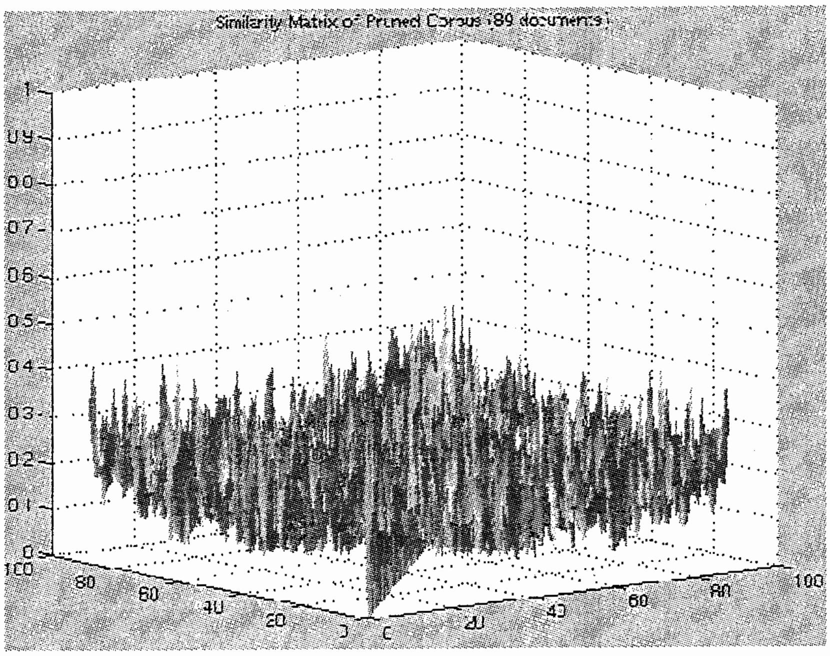
The graph below compares weight vectors for two sets of documents.



The cross-similarity matrix, containing SIM(d(i),d(j)) for each unique pair of documents in the corpus, is shown below for a set of 150 documents.



Similarity matrix of the corpus pruned with a threshold value 0.5 is shown below.


Similarity Matrix of Pruned Corpus (89 documents)
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
100
80
60
40
20
0
0
20
40
60
80
100

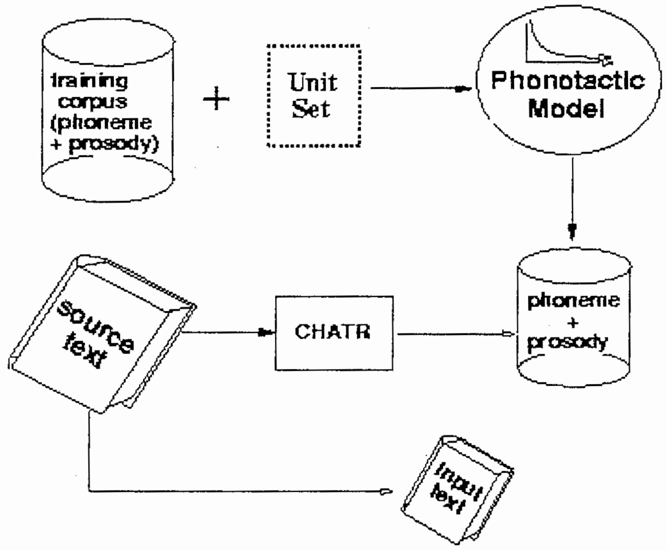
Automatic Database Recording and Balancing Tool

Although the input text can be pre-compiled off-line, the balancing process does not stop here. Before a CHATR database is produced, the text must be read, recorded, labeled with phonetic and prosodic tags, indexed and compiled into database format.

Up to the recording, the phonetic and prosodic properties of the database are predicted and the balance is pre-determined. However, the human interaction introduces new information and variability into the unit distribution. The sources of variability are, for example:

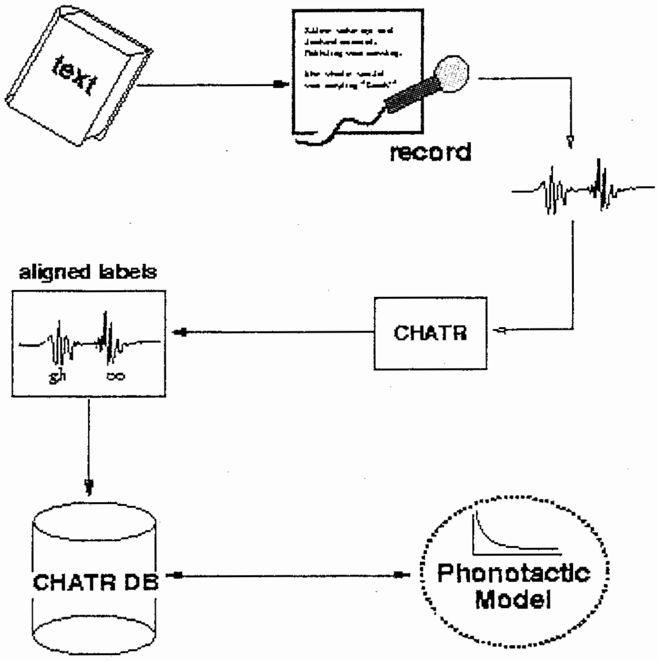
- pronounciation of words (there are 80 pronounciation variants of the word "and" in the Switchboard corpus)
- insertion and omission of pauses
- prosodic contour of the utterance (stress or no stress, etc.)

Therefore, it is desirable to analyse the unit balance of the resulting waveform and compare it to the originally intended model. If there are significant deviations from the model, we might want to re-insert some of the missed units by having the speaker read more text. We would also like to automate the entire process according to the system architectural diagrams below.



The Off-Line System

- a phonetic corpus is taken to be the "universe"
- the phon. model is trained on the corpus w/respect to the unit set
- phonetics and prosody (pre/post boundary, +/- stress) are predicted for a source text using CHATR
- the text is reduced using a balancing algorightm



The On-Line System

- the user is promted with sentences from selected text
- user has control over the timing and editing of wave file
- possible pronounciation variants and prosidic contours are predicted
- choosing the best match, align labels with the wave file
- after the text is recorded, analyse and reduce the units based on similarity
- compare database distribution with the original model

Conclusion

As a result of the research conducted, we suggest that recording a balanced database means

1. automatically extracting a multi-phone unit set from target language based on Maximum Likelihood
2. using phonotactic modelling of the language to predict what speech units are likely to be synthesised.
3. generating input text using similarity to the model as a measure of balance
    1. accumulate text items by greedy strategy
    2. cover limited subset of units
4. automatically learning statistics about the recorded database to assess actual balance of units.

The next step is implementing a system that combines these methods into an interactive recording tool.