Conversing with Stochastic Language Models

Jason L. Hutchens
Artificial Intelligence
hutch@a-i.com

Abstract

In this paper we look at the problem of conversation simulation via stochastic language modelling. In particular, we show how a stochastic language model may be used to discover constraints which are beneficial when using the model generatively in the context of conversation simulation, we introduce a goal-oriented language model which is conducive to having its generations constrained in this way, and we discuss how the UpWrite, a technique borrowed from the field of syntactic pattern recognition, may be used to abstract the model's representation of user input by discovering higher-level structure in the observed data.

1 Introduction

Conversation simulation is a term coined to describe the ultimate goal of chatbots, computer programs which engage the user in a dialogue. Chatbots have traditionally relied on rather *ad hoc* techniques, such as hard-wired keyword-based rules and template matching, to achieve the goal of conversation simulation.

In this paper we explore some stochastic language modelling techniques which may enable the development of conversations simulators which are more autonomous, and freer of assumptions, than the norm. To drive home the power of the methods which we will be discussing, it should be noted that we shall not even make the traditional *a priori* assumption that user input will be in the form of sentences in the English language.

Throughout this paper we shall, where appropriate, denote whitespace with the $\wedge$ symbol.

2 Stochastic Grammatical Inference

Markov models may be trivially inferred from data, after which they may be turned on their heads and used *generatively*. This is nothing new, Shannon himself gave an example generation in his famous 1948 paper, and we reproduce Shannon's example in figure 1 (Shannon and Weaver, 1949).[1]

It is apparent that the generated data, although gibberish, is both novel, in that it does not appear in the data used to infer the model, and is locally grammatical. This, and the fact that Markov models embody both the context-free and context-sensitive redundancy which is present in natural languages, and which has been suggested to be a necessary feature of all complex systems (Campbell, 1984), suggests that we may reasonably inquire how far such simple modelling techniques may be taken, and whether they are applicable to the problem of conversation simulation.

> THE HEAD AND IN FRONTAL ATTACK ON AN ENGLISH WRITER THAT THE CHARACTER OF THIS POINT IS THEREFORE ANOTHER METHOD FOR THE LETTERS THAT THE TIME OF WHO EVER TOLD THE PROBLEM FOR AN UNEXPECTED

Figure 1: Data generated by Shannon in 1948 using, in effect, a 1^{st}–order Markov model.

Before progressing down this winding path of investigation, we introduce some definitions, generalising beyond Markov models, to predictive models, along the way.

Definition 1 *An* alphabet *is a finite set containing at least two distinct elements known as the* symbols. *Let $\mathcal{A}$ represent the alphabet, let $x \in \mathcal{A}$ denote some symbol from the alphabet, and let $|\mathcal{A}|$ denote the cardinality of the alphabet.*

[1] Note that Shannon approximated the behaviour of a 1^{st}–order Markov model by writing down a word, flipping through a book at random until he encountered that word again, writing down the word which followed it, and repeating.

Definition 2 *Let $s_z = x_1, \ldots, x_i, \ldots, x_z$, with $x_i \in \mathcal{A}\ \forall\ x_i$, be a sequence of z symbols which we shall refer to as a* symbolic time series*, the* data*, or a* corpus.

Definition 3 *A* stochastic language model *is a model which is capable of assigning a probability $P(s_z)$ to an arbitrary symbolic time series $s_z = x_1, x_2, \ldots, x_z$.*

Remark 1 *It is common to decompose the probability $P(s_z)$ using Bayes' rule, as in equation 1, where $s_{i-1} = x_1, x_2, \ldots, x_{i-1}$ is the* history*. It should be noted that this decomposition is not unique—there are many possible decompositions, all of them equally valid.*

$$P(s_z) = \prod_{i=1}^{z} P(x_i|s_{i-1}) \tag{1}$$

Definition 4 *A* predictive model $\mathcal{M}$ *is a stochastic language model which is capable of making a prediction about the next symbol x_i in the symbolic time series s_z in the form of a probability distribution over the alphabet. We denote the probability which the predictive model assigns to the next symbol in the data as $P(x_i|\mathcal{M}, s_{i-1})$.*

Remark 2 *The inference problem for predictive models is therefore one of estimating the conditional probabilities of the right-hand side of equation 1. This is difficult as i increases, as it becomes more and more likely that the history s_{i-1} will not have been observed in the training corpus at all. The common solution to this problem is to group histories into a finite set of* equivalence classes*.*

Definition 5 *An* equivalence class of strings *is a set of strings which are deemed to be similar according to some measure. We use $\Phi(s_{i-1})$ to denote the equivalence class to which the string s_{i-1} belongs. The decomposition of equation 1 may now be expressed as in equation 2. Our problem is now one of selecting an appropriate equivalence classification.*

$$P(s_z) \approx \prod_{i=1}^{z} P(x_i|\Phi(s_{i-1})) \tag{2}$$

Remark 3 *The equivalence classification traditionally used is to classify histories according to their most recent context of n symbols, as in equation 3. This is the* Markovian assumption*, and predictive models which use this equivalence classification are known as n^{th}–*order Markov models*.*

$$\Phi(s_{i-1}) = \langle x_{i-n}, \ldots, x_{i-1} \rangle \tag{3}$$

Remark 4 *The n^{th}–order Markov model $\mathcal{M}$ estimates the probability of the data s_z as in equation 4.*

$$P(s_z|\mathcal{M}) \approx \prod_{i=1}^{z} P(x_i|\langle x_{i-n}, \ldots, x_{i-1} \rangle) \tag{4}$$

Remark 5 *The maximum-likelihood estimate of the probabilities of the right-hand side of equation 4 are derived from the normalised frequency with which the substring $x_{i-n}, \ldots, x_i$ occurs in the training corpus. This is achieved as in equation 5, where $C(s_i)$ denotes a count of the number of occurrences of the substring s_i in the training corpus.*

$$P(x_i|\langle x_{i-n}, \ldots, x_{i-1} \rangle) \approx \frac{C(x_{i-n}, \ldots, x_i)}{C(x_{i-n}, \ldots, x_{i-1})} \tag{5}$$

3 Learning to Talk

Stochastic language models may be used generatively, as we have shown, but this does not a conversation simulator make. Learning a language and learning to use it are two different processes, and we consider the latter equivalent to learning how to properly *constrain* the stochastic language model during generation. The introduction of constraints, if done carefully, allows the generated data to become relevant, in some limited way, to whatever task the stochastic language model is being applied to.

The simplest form of constraint would be the requirement that the generated data contain some number of pre-determined symbols.[2] This is not to say that constraints can only take the form of symbols; they may equally well come from the environment (information about objects in the room, for example), provided that we are able to specify a method of constraining the language model using this information.

The problem we now face is one of automatically determining what the constraints should be, and incorporating them into the predictive model. Information theoretic measures have proved themselves to be useful in this respect.

3.1 Information Theory

Information theory allows us to quantify the intuitive notions of *surprise* and *uncertainty*, relative to a predictive model and a symbolic time series. We shall show later how these two measures may be used to discover structure in the data which lies just beyond the reach of the model; for now, we satisfy ourselves with showing how information theoretic measures may be used to find interesting symbols

[2] For example, the requirement that the generation contain the symbol `hello`.

in a novel sequence of symbols, such as a sentence entered into our conversation simulator by the user. We begin with definitions.[3]

Definition 6 *The* information supplied to the predictive model by the next symbol in the data *is denoted* $I(x_i|\mathcal{M}, s_{i-1})$*, where* x_i *is the symbol which follows the history* s_{i-1} *and* $\mathcal{M}$ *is the predictive model, and is given by the negative logarithm, taken to base 2, of the probability of the symbol* x_i *following the history* s_{i-1} *according to the predictive model* $\mathcal{M}$*, that is,*

$$I(x_i|\mathcal{M}, s_{i-1}) = -\log_2 P(x_i|\mathcal{M}, s_{i-1}) \qquad (6)$$

Remark 6 *The information supplied to the predictive model by the next symbol in the data specifies the minimum number of bits required to describe the symbol with respect to the model in an unambiguous way, and may be understood informally to represent the* surprise *the predictive model receives upon discovering what the next symbol in the data actually is.*

Definition 7 *The* mutual information between two symbols *is denoted* $\hat{I}(x_a; x_b|\mathcal{M})$*, where* x_a *and* x_b *are the two symbols under consideration, and* $\mathcal{M}$ *is a stochastic language model, and is given by the net reduction in information provided to the model by the event* $\langle x_a; x_b \rangle$ *following observation of one of the two symbols of the pair, that is,*

$$\hat{I}(x_a; x_b|\mathcal{M}) = \log_2 \frac{P(x_a, x_b|\mathcal{M})}{P(x_a|\mathcal{M})P(x_b|\mathcal{M})} \qquad (7)$$

Remark 7 *The mutual information between two symbols specifies the amount of information the model will receive upon observing one of the symbols given that it has already observed the other and, as such, provides a good measure of the correlation between the two symbols, with a low value implying that the symbols are strongly correlated.*

Definition 8 *The* instantaneous entropy of the predictive model *is given by the expected value of* $I(x_i|\mathcal{M}, s_{i-1})$*, and is denoted* $H(\mathcal{M}, s_{i-1})$*, where* s_{i-1} *is the history and* $\mathcal{M}$ *is the predictive model, that is,*

$$H(\mathcal{M}, s_{i-1}) = \sum_{x_i \in \mathcal{A}} P(x_i|\mathcal{M}, s_{i-1}) I(x_i|\mathcal{M}, s_{i-1}) \qquad (8)$$

[3]Note that our terminology differs slightly from that used by Shannon, in that we consider information to be an *a posteriori* property of data with respect to some model, and we define the entropy as an *a priori* property of the model itself.

Remark 8 *We intuitively think of* $H(\mathcal{M}, s_{i-1})$ *as the* uncertainty *the predictive model* $\mathcal{M}$ *has about the next symbol in the data, given that* s_{i-1} *is the history.*

3.2 Finding Interesting Symbols

A predictive model may locate symbols in a symbolic time series which are, to it, interesting, in the sense that they convey the greatest amount of information. The *MegaHAL* conversation simulator employs this method to generate semi-appropriate, and occasionally extremely apt, replies to user input (Hutchens, 1998).

As an example, consider the first three sentences of the Sherlock corpus, shown in figure 2 (Doyle,). A word-level 1^{st}–order Markov model inferred from the entire Sherlock corpus experiences the most surprise upon encountering the emphasised words.

> *To* Sherlock Holmes *she* is always the *woman.* I have seldom heard him *mention* her *under* any other *name.* In his eyes she *eclipses* and *predominates* the whole of her *sex.*

Figure 2: The first three sentences of the Sherlock corpus, with the three most surprising words in each sentence, relative to a 1^{st}–order Markov model inferred from the entire corpus, emphasised.

3.3 Learning Input/Output Symbol Relationships

Information theoretic measures may be employed to discover symbol relationships between two symbolic time series, one representing user input to the program and the other representing appropriate output, by measuring the mutual information between each possible pairing of symbols, relative to an appropriate stochastic language model. For example, the model may learn that if the symbol `why` appears in a question, the symbol `because` should appear in its answer. The *NonI* conversation simulator was developed with this ability in mind (Hutchens,).[4]

It should be noted that an "appropriate stochastic language model" is merely a model which is able to provide estimates of the probabilities required to calculate $\hat{I}(x_q; x_a|\mathcal{M})$, where x_q represents a symbol in the *question*, and x_a represents a symbol in the *answer* to that question. Evidently a variation on a

[4]This technique makes an *a priori* assumption that the corpus used for training contains a dialogue. As large corpora of this type are not readily available, training is typically performed online, although inference of the stochastic language model used for generation may still be performed offline from large natural language corpora.

standard 1^{st}–order Markov model is sufficient here (with the difference being the context used).

3.4 Introducing MegaHAL

The MegaHAL conversation simulator, discussed in a paper presented at the 1998 Human-Computer Conversation workshop (Hutchens, 1998), finds the most "interesting" symbol in the user's input using the aforementioned technique, transforms it using *a priori* information about the English language, and generates output constrained on the symbol which results.

The generations of a Markov model are necessarily a random walk, and constraining such models to only generate sequences which contain a symbol specified *a priori* is difficult. The solution implemented in the MegaHAL conversation simulator was to use *two* Markov models, inferred from the same data, but operating in opposite directions. Generation begins at the constraining symbol, and proceeds from there towards the start of the sentence in one direction and towards the end of the sentence in the other.

Surprisingly, perhaps, users of MegaHAL unashamedly anthropomorphise the program, often laying claims to actual intelligence. Our favourite example of this phenomenon is that of the clergyman who spent hours teaching MegaHAL about the love of Jesus only to receive blasphemous responses, which spurred him on more and more. Human beings have a natural tendency to perform pattern recognition, with the result that users of MegaHAL are prepared to read meaning into its replies where none exists, forgiving 99% of gibberish generations if one in a hundred are astonishing.

4 Novel Stochastic Language Models

Generating data subject to constraints is difficult using traditional Markov models, so we shall now consider the development of a new form of stochastic language model which is conducive to having its generations constrained by a template of several predetermined symbols.

4.1 The Fractal Language Model

The *fractal language model* is termed thus due to the recursive nature of its generation process. Statistics are collected about symbols which occur anywhere between two "anchor symbols", separated by an arbitrary distance, as in equation 9. Generation begins with a template, which contains some number of constraining symbols, and the language model recursively "fills in the gaps" in this template. Generation continues until some pre-determined stopping criterion is met.[5]

$$P(x_j|\mathcal{M}_{fractal}, x_i, x_k) \approx \frac{C(x_i, \ldots, x_j, \ldots, x_k)}{C(x_i, \ldots, x_k)} \quad (9)$$

In figure 3 we show three of the more successful generations of a fractal language model, operating on the word-level, inferred from a corpus of 174 sentences extracted from the Probert E-Text Encyclopaedia, Edition 10.0 (Probert,). These generations were constrained by a template containing a pair of constraining symbols which had been previously observed together in at least one of the training sentences, and they exhibit long-distance dependencies, a desirable feature of any stochastic language model. For example, in the second sentence, the symbols `Dance` and `movement` are clearly related, even though they are widely separated, and these two symbols were, in fact, the constraining symbols in the template of this generation.

> A sword is an offensive weapon designed chiefly for the sale and consumption of deep unconsciousness.
>
> Dance is a person who suffers from the carcass of movement.
>
> Cement is information, especially that stored in a small, usually tree dwelling primate.

Figure 3: Three sentences generated from a constraining template of two words by a fractal language model inferred from a small corpus of encyclopaedic information.

Various *ad-hoc* techniques were explored to further constrain the generations of the fractal language model, in the hope of avoiding the problems of deep recursion during generation, and the net result of this effort was the development of a new kind of Markov model.

4.2 The Goal-Oriented Language Model

The *goal-oriented language model* is a form of fractal language model which constrains the location of

[5]The stopping criterion used is to continue generation until all adjacent symbol-pairs in the generated sentence have been observed at least once in the training corpus. A major problem with this methodology is that symbol pairs such as `the-the`, in the case where English text is being modelled on the word-level, are never adjacent, and that the word `the` occurs between this pair with a high probability, and is therefore likely to be generated by the model.

generated symbols to immediately adjacent to the leftmost anchor symbol, as in equation 10. It is clear that this model is equivalent to a family of 1^{st}–order Markov models, each of which is constrained by a different symbol x_k occurring somewhere in the future.

$$P(x_j|\mathcal{M}_{goal}, x_i, x_k) \approx \frac{C(x_i, x_j, \ldots, x_k)}{C(x_i, \ldots, x_k)} \qquad (10)$$

This insight prompted the development of a general n^{th}–order goal-oriented language model consisting of a family of n^{th}–order Markov models, each of which is constrained by a different "goal symbol". We refer to such models as $\mathcal{M}_n^{x_k}$, where n indicates the order of the model, and x_k indicates the constraining symbol, which may be thought of as the goal which the model will head towards when used generatively.[6] We use the notation $\mathcal{M}_n^{\star}$ to refer to the standard, unconstrained, n^{th}–order Markov model.

Inference of an n^{th}–order goal-oriented language model is a simple process. For each prefix string s_k of the training data s_z, we update the statistics of model $\mathcal{M}_n^{x_k}$ by performing inference on the string s_{k-1} using regular Maximum Likelihood techniques.

Definition 9 *We define a string s_k to be a* prefix string *of a string s_n if the first k symbols in each of the two strings match, and $n \geq k$.*

In figure 4 we show five mildly amusing generations of a 1^{st}–order goal-oriented language model inferred from the same training corpus as in our earlier example and constrained in the same manner. These generations exhibit the same long-distance dependencies as those produced by the fractal language model.

5 The UpWrite

Another novel stochastic language model we have developed is based on the concept of the *UpWrite*, a generalised process for constructing hierarchical representations of data, developed by Michael Alder as an alternative to King Sun Fu's programme of syntactic pattern recognition (Alder, 1994; Hutchens, 1999).

The UpWrite works by iteratively extracting higher-level structure from the representation of the data at one level of the hierarchy in order to determine how the data should be represented at the next level of the hierarchy, with similarities between local models of the data at a particular level of representation being used to extract this structure. Two major types of structure are presumed by the UpWrite: *Sub-objects* are ordered sets of objects, while *quotient-objects* are equivalence classes of objects.

[6] Traditional Markov models generate sentences in a random-walk fashion, stumbling upon the end of the sentence by chance. Goal-oriented language models, however, generate sentences via a constrained random walk; they are aware of certain goal symbols through which they must pass on their way to the end of the sentence.

An acronym is a curved wooden weapon of other words.

A symbol is a suspended brass disk which represents something else.

Bone is a hollow shell filled with the external coating of an animal.

Leonardo da Vinci was an Italian artist and expert in Kung Fu who popularised the martial arts in unpublished note books.

The nose is an English naturalist. He published his theory of smell.

Figure 4: Five sentences generated from a constraining template of two words by an n^{th}–order goal-oriented language model inferred from a small corpus of encyclopaedic information.

5.1 The Sub-Object UpWrite

The *Sub-Object UpWrite* is motivated by the fact that data may be partitioned into a collection of sub-objects such that these sub-objects correspond to the primitives of some higher-level representation of the data. For example, a set of pixels in an image may correspond to a line segment, while a set of characters in natural language text may correspond to a word.

Definition 10 *The* Sub-Object UpWrite *is a mapping $\langle x_1, \ldots, x_j \rangle \mapsto y$ between a symbol sequence $\langle x_1, \ldots, x_j \rangle$ of symbols taken from some alphabet $\mathcal{A}_1$ to a symbol y in some new alphabet $\mathcal{A}_2$ such that the observed data $s_z = x_1, \ldots, x_z$, $x_i \in \mathcal{A}_1$ can be represented as a higher-level symbolic time series $t_k = y_1, \ldots, y_k$, $y_i \in \mathcal{A}_2$, $k \leq z$.*

Definition 11 *The* Sub-Object DownWrite *is the corresponding inverse relation.*

5.2 The Quotient-Object UpWrite

The *Quotient-Object UpWrite* is motivated by the fact that symbols may be assigned to equivalence

classes such that these equivalence classes correspond to the primitives of some higher-level representation of the data. For example, a quotient-object of "things" in an image may correspond to the class of straight line segments, while a quotient object of words in natural language text may correspond to the class of verbs.

Definition 12 *The* Quotient-Object UpWrite *is a mapping* $\{x_1, \ldots, x_j\} \mapsto y$ *between an equivalence class of symbols* $\{x_1, \ldots, x_j\}$ *taken from some alphabet* $\mathcal{A}_1$ *to a symbol* y *of some new alphabet* $\mathcal{A}_2$ *such that the observed data* $s_z = x_1, \ldots, x_z$, $x_i \in \mathcal{A}_1$ *can be represented as a higher-level symbolic time series* $t_z = y_1, \ldots, y_z$, $y_i \in \mathcal{A}_2$.

Remark 9 *The* Quotient-Object DownWrite *is not unique, since it is not possible to determine which lower-level symbol* x *should serve as the DownWritten version of the symbol* y. *It is possible to generate possible DownWritten versions of* t_z *by selecting a symbol* $x_i \in \{x_1, \ldots, x_j\}$ *at random. The corresponding DownWritten version of* t_z, *which is not unique, has the property that its UpWrite is* t_z, *and the DownWrite therefore provides an indication of how well the UpWrite has captured salient features of the data being modelled.*

5.3 The UpWrite Predictor

Application of the UpWrite concept to stochastic language modelling results in the UpWrite Predictor, shown in figure 5, consisting of a chain of modules, each of which contains an UpWriter, for discovering structure in the data using the predictions made by the model in the previous module, an alphabet, for containing new symbols which correspond to the higher-level structure discovered, and a predictive model, inferred from the UpWritten data.[7] Note that predictions made by the model feed back into the UpWriter, allowing erroneous generalisations to be corrected. The power of the technique is the fact that the data is represented, automatically, at various granularities, and that applications may select a level of representation appropriate to the task at hand.

Construction of the UpWrite Predictor may proceed by specifying techniques for

- finding Sub-Objects and Quotient-Objects in the data from the model's representation of it;
- forming a new representation of the data, with respect to this newly discovered structure; and
- correcting erroneous generalisations.

Sub-Objects and Quotient-Objects may be found in data by applying information theoretic measures to the sequence of predictions made by the model, resulting in the twin techniques of agglutination and agglomeration, which elegantly solve the first of the three dilemmas listed above. It is outside the scope of the current discussion to offer solutions to the remaining two; this has been done elsewhere (Hutchens, 1999).

5.3.1 Agglutination

The information provided to a predictive model by a symbol in the data may be considered to be a measure of the degree of independence between this symbol and the one which immediately precedes it, with respect to the model, with low values of information signifying a high degree of correlation between the two symbols.

The *agglutination* process works by finding the most correlated symbol pair in the data, forming a new, higher-level Sub-Object symbol from that pair, UpWriting the data with respect to this new symbol, and iterating until some stopping criterion is reached.[8]

It is possible to view the Sub-Object structure discovered in the data by the UpWrite Predictor at all levels of representation simultaneously in the form of a dendrogram. In figure 6 we show a dendrogram over the first sentence of the Sherlock corpus relative to a character-level UpWrite Predictor inferred from the entire corpus. The hierarchical structure exhibited in this illustration forms a reasonable decomposition of the sentence into phrases and words, with a few obvious mistakes that should be correctable via the feedback mechanism inherent in the model.

5.3.2 Agglomeration

Two symbols may very well belong to the same class if the predictive model cannot distinguish between them on the basis of the predictions it makes about which symbols are likely to follow them. Information theoretic measures may be used to calculate the similarity between the predictions made by the model, expressed as probability distributions over the alphabet.[9]

The *agglomeration* process works by finding the two symbols which are the most difficult to tell apart

[7] The lowest-level module in the chain lacks both an alphabet and an UpWriter due to the fact that, at the lowest-level, the alphabet is assumed.

[8] The most basic stopping criterion being "continue until the highest-level representation of the data consists of a single symbol".

[9] We neglect to specify candidate similarity measures here, as the measure chosen is irrelevant as far as the agglomeration method is concerned.

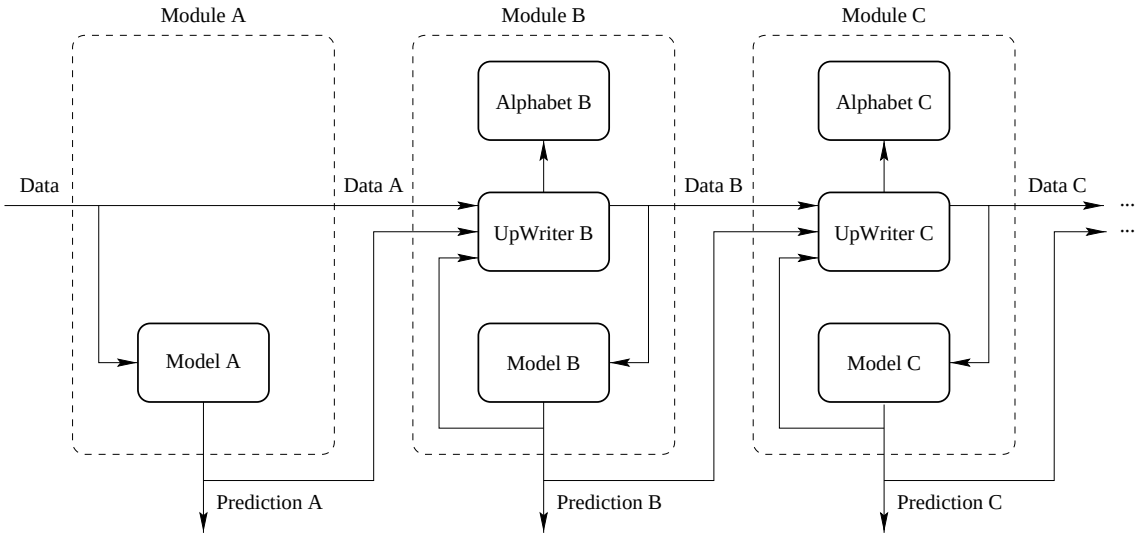

Figure 5: The structure of the UpWrite Predictor.

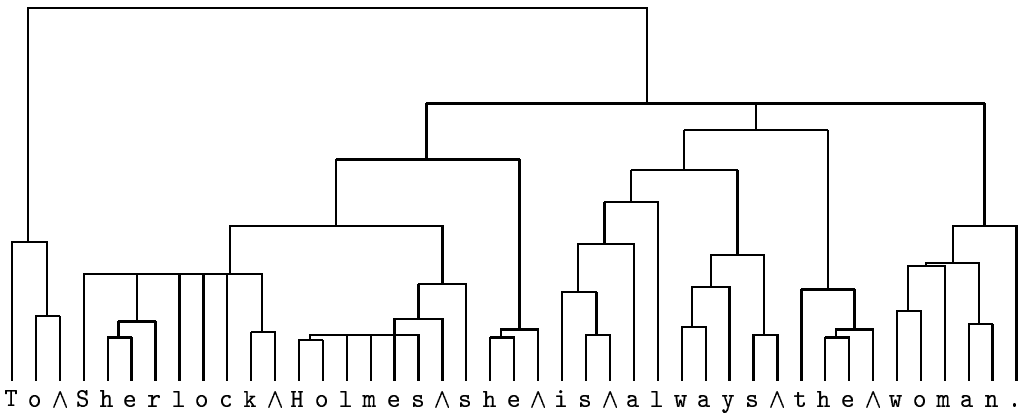

Figure 6: Dendrogram illustrating a hierarchy of Sub-Objects discovered by the agglutination process, at various levels of representation of the data, relative to a character-level UpWrite Predictor inferred from the Sherlock corpus.

according to this criterion, forming a new Quotient-Object symbol from the pair, UpWriting the data with respect to this new symbol, and iterating.

As with agglutination, it is possible to view, via a dendrogram, the Quotient-Objects found by the agglomeration process at all levels of abstraction simultaneously. Figures 7, 8 and 9 illustrate some of the hierarchical Quotient-Objects found by the algorithm, and it is evident that these Quotient-Objects are sufficiently fine-grained to be quasi-semantic in nature.

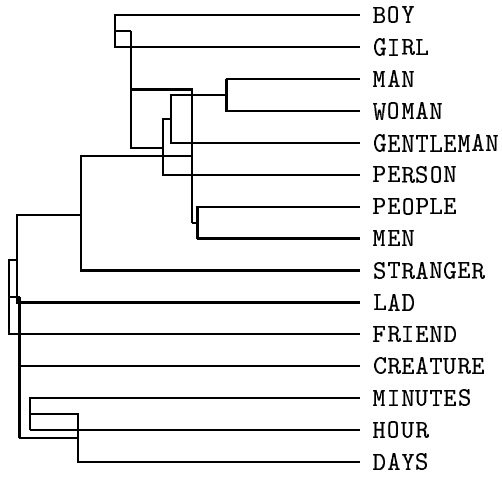

Figure 7: Dendrogram illustrating a hierarchy of Quotient-Objects discovered by the agglutination process, at various levels of representation of the data, relative to a word-level UpWrite Predictor inferred from the Sherlock corpus.

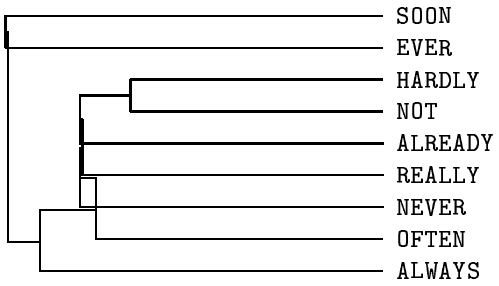

Figure 8: Dendrogram illustrating a second hierarchy of Quotient-Objects discovered by the agglutination process, at various levels of representation of the data, relative to a word-level UpWrite Predictor inferred from the Sherlock corpus.

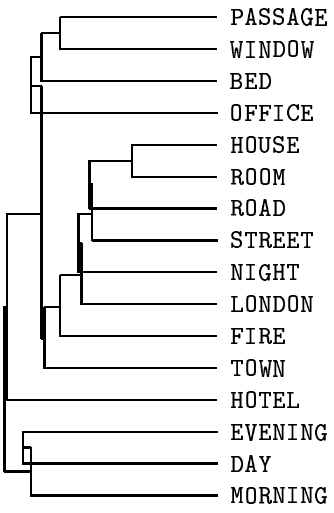

Figure 9: Dendrogram illustrating a third hierarchy of Quotient-Objects discovered by the agglutination process, at various levels of representation of the data, relative to a word-level UpWrite Predictor inferred from the Sherlock corpus.

5.4 Experimental Results

The UpWrite Predictor constructs a hierarchical representation of the data which it is inferred from, and this high-level representation may be DownWritten to produce an approximation to the original data, providing, if nothing else, a useful indication of the performance of the UpWrite Predictor, as it enables us to examine the structure extracted from the original data by eye.

A word-level UpWrite Predictor was inferred from the Sherlock corpus, and the UpWritten form of the first two sentences of that corpus was DownWritten in five different ways, as in figure 10.[10] For the purpose of comparison, recall that the first three sentences of the Sherlock corpus were shown in their original form in figure 2. This example immediately suggests another way of constraining the generations of the stochastic language model—constraints may be imposed by specifying a high-level (some may even say "concept-level") representation of the data, and allowing the model to, possibly in a constrained way, produce a lowest-level version of this representation.

One property of all of the five DownWritten forms shown is that they share the same high-level representation with respect to the UpWrite Predictor. This property allows us to, for example, abstract the

[10] Multiple DownWrites are possible due to the fact that the Quotient-Object DownWrite is not unique.

exact form of text entered by the user, and, ideally, would enable us to cluster these high-level forms into groups representing the same *concept*.

As a second example of generation using the UpWrite Predictor, consider figure 11, which shows a portion of data generated by a character-level UpWrite Predictor inferred from the Sherlock Corpus. Note that although the UpWrite Predictor incorporates 1^{st}–order Markov models only, the generation exhibits some features, such as the sequences `at^the^first^note`, `she^would^hardly^be` and `he^could^find^that^you`, that we would normally not expect to see in the generations of a character-level, 1^{st}–order Markov model.

5.5 Potential for Conversation Simulation

The UpWrite technique has many properties which make it attractive to the designer of a conversation simulator, including

- an ability to discover words, phrases and syntactic (even quasi-semantic) word classes automatically;
- a simple procedure for generating novel data; and
- a potential to abstract the lowest-level representation of a sentence to the level of "concepts".

We postulate that the UpWrite Predictor, in combination with other components, such as a component for learning and generating appropriate constraints for sentence generation, may form the genesis of a conversation simulator which makes few *a priori* assumptions about the world, and which may be incrementally trained from natural language dialogues. The development of such a system has begun, and we are full with anticipation of what the next few years will bring.

6 Summary and Conclusion

In this paper we have introduced various stochastic language modelling techniques which may prove fruitful to the practitioner insofar as conversation simulation is concerned. It is our hope that the methods described herein will, in combination with other sophisticated techniques, form the basis of a generalised language acquisition system.

To Sherlock Holmes he was always the woman. I have seldom heard me mention her on any other name.

To Sherlock Holmes who it is always to from of the woman but I had seldom heard myself mention her under any other name.

To Sherlock Holmes she it was always to with of the woman, and I have seldom heard myself mention her through any other name.

To Sherlock Holmes he is always a woman. I had seldom heard me mention her on any other name.

To Sherlock Holmes she was always to from the woman, and I have seldom heard us mention her by any other name.

Figure 10: Most probable, least probable, longest, shortest and random DownWrites of the same higher-level representation of the first sentence of the Sherlock corpus, relative to a word-level UpWrite Predictor inferred from that corpus.

do not in the eletely at the first note in is nothing of For in your hair, Watson! Godget of the tlances Mr. He's ink-cl?" "The ses muchying! This about Grun to she would hardly be hoprightly visized glookso son that up in the doingle in my find-gun to resh he could find that you but unny our very kind. This could reater which led er in his beach was no silebject the cowr." Se, when the young Murn only one by this ushed

Figure 11: Data generated by a character-level UpWrite Predictor inferred from the Sherlock corpus.

References

Alder, Michael D. 1994. Inference of syntax for point sets. In E. S. Gelsema and L. N. Kanal, editors, *Pattern Recognition in Practice IV: Multiple Paradigms, Comparative Studies and Hybrid Systems*, number 16 in Machine Intelligence and Pattern Recognition, pages 45–58. Elsevier Science B.V., June.

Campbell, Jeremy C. 1984. *Grammatical Man: Information, Entropy, Language and Life.* Pelican Books.

Doyle, Sir Arthur Conan. The Sherlock/SHERLOCK corpora. These corpora were formed by Jason Hutchens from a collection of texts downloaded from the Oxford Text Archive, and are available from:
`http://ciips.ee.uwa.edu.au/~hutch/sherlock/`

Hutchens, Jason L. NonI: A non-intelligent conversation simulator. Technical presentation given as part of the CIIPS seminar series. Available at:
`http://ciips.ee.uwa.edu.au/~hutch/phd/talk6.ps.gz`

Hutchens, Jason L. 1998. Introducing MegaHAL. In David M. W. Powers, editor, *NeMLaP3 / CoNLL98 Workshop on Human-Computer Conversation, ACL*, pages 271–274, January.

Hutchens, Jason L. 1999. *The UpWrite Predictor: A General Grammatical Inference Engine for Symbolic Time Series with Applications in Natural Language Acquisition and Data Compression.* Ph.D. thesis, Department of Electrical & Electronic Engineering, The University of Western Australia, Australia 6907. Available from:
`http://ciips.ee.uwa.edu.au/~hutch/phd/`

The Probert e-text encyclopaedia. Available at:
`http://www.pins.co.uk/upages/probertm/`

Shannon, Claude E. and Warren Weaver. 1949. *The Mathematical theory of Communication.* University of Illinois Press.