Problems in Semitic NLP: Hebrew Vocalization Using HMMs

Leonid Kontorovich
Princeton University
Supervisor: Daniel Lee

Abstract

Semitic languages present a special problem to Natural Language Processing in that most of the vowels are omitted from written prose. Thus a natural (and often necessary) preprocessing step for Semitic text is vocalization (or "pointing") of the text: filling in the vowels and diacritical marks. Unvocalized Hebrew contains many more ambiguities than English text; conversely, vocalized Hebrew is considerably less ambiguous than English. The vocalized text could then be fed into parsers, translators, text-to-speech engines, or simply taken as the end product (for young children and beginners). This paper demonstrates that Hidden Markov Models may be a useful tool for vocalization, and evaluates some existing approaches to this problem.

1. Introduction

Natural Language Processing is largely aimed at automating various language related tasks that are presently performed by humans. These include intelligent searching (as opposed to simple pattern matching), text-to-speech, speech recognition, to list a few. One of the greatest obstacles to processing language is its inherent ambiguity. Human speakers and readers rarely encounter this obstacle (and when they do, it often becomes a source of amusement). Because of priming, contextual analysis, a feel for likelihood, and other processes that are not well understood, humans manage to ignore the "irrelevant" possible interpretations of an utterance and tend to zero in on a single one. If an automated system is to execute the intended command, it too must choose the "right" interpretation out of a number of possibilities.

It so happens that Semitic written languages have an additional ambiguity that roughly amounts to the vowels being omitted from the text. This understandably increases the number of interpretations an utterance could have. Conversely, when all the vowel-markers are included in the text, it becomes a good deal less ambiguous than English.

It is thus a very natural preprocessing step to supply the vowel-markers to Semitic text. It would certainly aid in parsing and translation, and seems indispensable in text-to-speech. Since beginners find it easier to have the vowel-markers, adding them to the text need not be merely a preprocessing step but may be taken as a goal in itself.

Let us briefly define *morphology* as the system of deriving related words from a root. We observe that the English verb system has a relatively simple morphology: a typical verb may only take four inflected forms ("walk", "walks", "walked", "walking"). It is thus quite feasible to build a dictionary of English verbs in all possible conjugations. In other languages, however, a verb can take on hundreds of inflected forms. This makes it impractical to store every verbal form in a dictionary, and necessitates a morphological analyzer.

The Indo-European languages have a morphology that is markedly different from the Semitic languages. I.E. morphology may be called *linear* (or *concatenative*), in the sense that words are derived from a root by affixing prefixes and suffixes to the basically unchanging stem. (To cite an example from English, observe how the root **solu-* gives rise to *soluble*, *solution*, *dissolve*, *resolve*, etc.) New words are coined and derived according to this principle (e.g., *pasteurize*, *pasteurization*).

The Semitic languages, on the other hand, have a morphology that has been called *nonlinear* (or *nonconcatenative*). The root consists of a group of consonants (usually three or four), and depending on the inflection paradigm, different vowels are inserted between these root consonants.

We illustrate this point with an example from Hebrew. The group of consonants אכל (ʾKL) corresponds to the basic notion of "eating".

One can then derive various forms from this root, such as:

he eats	[ʾokhel]	אוֹכֵל
he is eaten	[neʾekhal	נֶאֱכָל
he dissolves	[meʾakel]	מְאַכֵּל
he is dissolved	[meʾukal]	מְאֻכָּל
he feeds	[maʾakhil]	מַאֲכִיל

he is fed	[mo'akhal]	מָאָכָל
he is dissolved (refl. sense)	[mit'akel]	מִתְאַכֵּל

Note that the consonants אכל persist while the pointing (vowels) varies.

When new roots are introduced into the verbal system, they must be cast in the Hebrew paradigm. Thus, to make *pasteurize* a Hebrew verb, one takes the consonants פסטר (PSTR) and supplies the pointing according to the inflection:

פִּסְטוּר	מְפַסְטֵר	לְפַסְטֵר
[pistur]	[mephaster]	[lephaster]
pasteurization	he pasteurizes	to pasteurize

Vocalization (or *pointing*) refers to the diacritical marks used to indicate vowels, geminations, and other specifics of Semitic text[1]. Written Semitic prose is ordinarily unpointed. The crux of the problem explored herein is to convert unpointed text into pointed text.

2. Motivation

Henceforth we shall focus on Hebrew as a typical example of a Semitic language.
It has been remarked that pointing Hebrew text is a natural disambiguation step, quite useful both for further machine processing (translating, text-to-speech, etc) and for human readers (children, beginners).
But pointing may also be studied on its own as an interesting problem in computational linguistics. It provides a precise and objective method to gauge the accuracy of our language models. Unlike parsing, text-to-speech, and translation, where rating the performance of an algorithm is a challenge in itself, pointing algorithms may be judged using very simple criteria (such as the percentage of correctly pointed words). Additionally, there is no need for extensive human preparation of a corpus. Assuming that large corpora of pointed Hebrew already exist, no further tagging of the text is required. This holds true both for training and evaluating the algorithm. Finally, even if one has no access to the methods of an algorithm, one can set very tight bounds on its performance using only a few carefully chosen examples. Later in this paper, we shall evaluate a commercial software package precisely in this way.

3. How difficult is the problem?

It is natural for a non-speaker of a Semitic language to inquire just how difficult this problem is for a human. It is no controversial matter, for example, that it is much easier for a human to read a text aloud (do text-to-speech) than to translate it into a different

[1] Although *vocalization* seems to be the term more accepted in the literature, it is often confused with text-to-speech. This paper will use the two interchangeably, preferring the more transparent and descriptive term *pointing*.

language. Where does the difficulty of vocalization fall on this scale of linguistic operations?

To get a very rough feel for the process, an English speaker might take a text sample, replace some of the vowels (say, the *o*'s and the *u*'s) with place-holders (such as the symbol #), remove all other vowels, and attempt to read the text. Informal experiments indicate that though English speakers understandably have a harder time reading such text, they can almost always restore the missing vowels. Strictly speaking, these experiments with English are in no way indicative of the difficulty of pointing Hebrew text – these are two entirely unrelated languages[2]. However, the fact that Hebrew is "designed" to be written unvocalized and that Hebrew speakers are trained to read unpointed text from a young age indicates that if anything, the problem of restoring the vowels ought to be easier in Hebrew than in English. We remark that an Israeli reads Hebrew text[3] just as quickly and efficiently as an American does English, making analogous mistakes (e.g., in English, words such as *read* and *lead* may be read wrong with insufficient context). It is the context that allows the readers to read with a high accuracy, and we shall attempt to capture some contextual relationships in our model.

4. Existing software

Given the demand for automated pointing, one might expect to find various approaches implemented in existing software. We were able to locate one commercial package, called *Nakdan Text*, produced by the Israeli Center for Educational Technology (המרכז לטכנולוגיה חינוכית). The product manual claims that "the program points the entire text according to morphological analysis […] and contextual analysis" to an accuracy of 90-95%. Without being able to obtain any information on the methods used in *Nakdan*, we purchased the software and conducted a few tests on it.

Contextual analysis is a far-reaching process. Hebrew is a highly inflected language that demands various grammatical agreements (gender, number) between its parts of speech. Although we examined numerous examples where a potentially ambiguous sentence may only receive one grammatical vocalization, we shall only present here a single line of query. We tested Nakdan's ability to observe the gender (masculine/feminine) agreement. The results presented here are typical of our systematic analyses of other grammatical aspects. We discuss 8 examples below:

[2] At least according to the commonly accepted modern theories.

[3] We note that Hebrew newsprint is almost entirely unpointed, novels and literature have partial pointing for ambiguous/unusual/foreign words, and poetry is almost fully pointed.

(*) indicates an ungrammatical construction
the underlined endings must agree (be identical) for grammaticality
note: the word order in each case is grammatical for Hebrew

1. NOUN-fem VERB-fem VERB-inf
   "woman tries to-dance"

   [isha menasa lirkod] אִשָּׁה מְנַסָּה לִרְקֹד.

2. * NOUN-fem(irr) VERB-masc VERB-inf
   "bird tries to-fly"

[tzipor menase lirkod] צִפּוֹר מְנַסֶּה לָעוּף.

3. * NOUN-fem(irr) ADJ-fem VERB-masc VERB-inf
   "bird pretty tries to-fly"

[tzipor yafa menase lirkod] צִפּוֹר יָפָה מְנַסֶּה לָעוּף.

4. * NOUN-fem ADJ-fem VERB-masc VERB-inf
   "woman pretty tries to-dance"

[isha yafa menase lirkod] אִשָּׁה יָפָה מְנַסֶּה לִרְקֹד.

5. * NOUN-fem ADJ-fem CONJ ADJ-masc VERB-masc VERB-inf
   "woman pretty and-foolish tries to-dance"

[isha yafa ve-shote menase lirkod] אִשָּׁה יָפָה וְשׁוֹטֶה מְנַסֶּה לִרְקֹד.

6. NOUN-fem PRO-COPULA-fem ADJ-fem
   "the-woman she pretty" = "The woman is pretty."

[ha-isha hi yafa] הָאִשָּׁה הִיא יָפָה.

7. * NOUN-fem PRO-COPULA-fem ADJ-fem CONJ ADJ-masc
   "the-woman she pretty and-foolish" = "The woman is pretty and foolish."

[ha-isha hi yafa ve-shote] הָאִשָּׁה הִיא יָפָה וְשׁוֹטֶה.

8. * NOUN-fem PRO-COPULA-fem ADV ADJ-masc
   "the-woman she very pretty"

[ha-isha hi me'od yafe] הָאִשָּׁה הִיא מְאֹד יָפֶה.

First let us focus on examples 1-3. Sentence 1 suggests that *Nakdan* respects gender agreement between nouns (אִשָּׁה) and verbs (מְנַסָּה). But in sentence 2, an irregular feminine noun (צִפּוֹר) gets a masculine verb (מְנַסֶּה). One might conclude that *Nakdan* is not aware that this noun (which doesn't have the typical Hebrew fem. ending) is feminine. But no – in sentence 3, we see that it gets a feminine adjective (יָפָה). One would be hard pressed to come up with a logically consistent set of heuristics to explain this behavior.

Examples 4-8 indicate that the contextual analysis spans at most two adjacent words, and fails to do this consistently. Consider sentence 4. We have a feminine subject (אשה), its attributive adjective (יפה) and its predicate verb (מנסה) as the first three words of the sentence. The adjective gets the feminine vocalization while the verb already does not; *any* two-word contextual analyzer would prohibit a verb from being pointed as מְנַסָּה when the preceding word is יָפָה.

Examples 5-8 specifically examine the effect of conjunctions and adverbs. In sentence 5, when two adjectives modify a feminine noun, only the first is pointed feminine. Both the second adjective and the verb have the masculine vocalization. Sentence 6 shows that gender agreement is observed for pronouns – the adjective is correctly feminine (here, the pronoun acts as a copula). In sentence 7, just as in 5, we see that a conjunction disrupts the contextual dependency. In sentence 8, an adverb has the same effect.

The previous eight examples examined *Nakdan*'s contextual analysis. We now turn to its morphological analyzer. For most ambiguous strings of consonants, *Nakdan* does indeed provide an exhaustive list of possible vocalizations, including those that native speakers would have difficulty coming up with. However, a cursory analysis indicates that the morphological derivations are not performed by a generative engine, but most likely through various lookup tables and heuristics. How else to explain the fact that *Nakdan* was thoroughly "familiar" with the root פסטר (pasteurize), in that it pointed the following correctly:

לְפַסְטֵר , מְפַסְטֵר , פִּסְטוּר

[pistur], [mephaster], [lephaster]

pasteurization, he pasteurizes, to pasteurize

but was not as consistent with קמפל (compile):

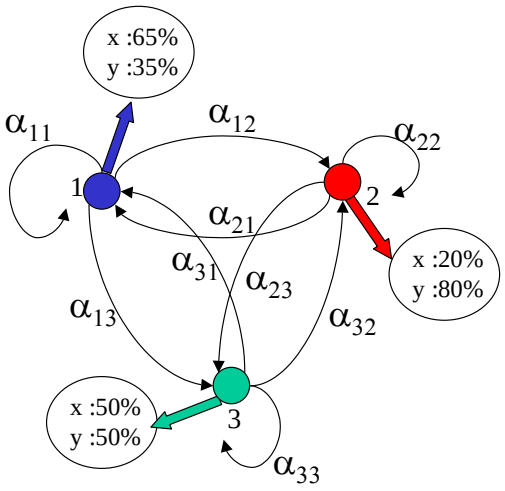
לְקַמְפֵּל , מְקַמְפֵּל , קמפול

where it failed to vocalize the underlined word although it fits the same exact paradigm as with the root פסטר. To use an analogy, it is as if an English morphological analyzer could handle "to compile", "he compiles", but not "compilation".

Before we leave *Nakdan*, let us asseverate that it is not our intent to disparage the effort that went into making it. The examples presented here were carefully chosen to illustrate theoretical points about *Nakdan*'s contextual and morphological limitations. Informal manual testing on newspaper articles showed that 90% is indeed a reasonable rating of *Nakdan*'s accuracy. This software embodies an important technical achievement and is without a doubt of considerable utility to anyone in need of pointing long Hebrew texts. We simply observe that the theoretical problems we set out to tackle have not yet been solved.

5. Review of HMMs

Since our methods are based on Hidden Markov Models, we briefly review the concepts and the mathematics involved. Underlying a Hidden Markov Model is a graph on N nodes (or, equivalently, a system that can be in N possible states – see figure). The states are indexed 1 through N.



$N = 3, M = 2$

[Figure 1. This system has 3 hidden states and 2 emission states]

The system starts in one of the N states, where the probability that it starts in state i is given by π_i (π is an $N \times 1$ matrix). The system then evolves according to the Markov condition, in which the state of the system at time t only depends on its state at time $t - 1$. This dependency is probabilistic; if the system is in state i at time t, then it will be in state j at time $t + 1$ with probability $\Pr(i \to j) = \Pr(j \mid i) = \alpha_{ij}$ (where α is an $N \times N$ matrix).

As the system evolves through T time steps, we shall index the states that it has taken at times 1 through T by $Q = q_1\ q_2 \ldots\ q_t\ \ldots\ q_T$. What gives this system the name *Hidden* Markov Model is that we do not actually observe the sequence of states Q. Instead, at each time step, the state i which the system has assumed emits one of M possible symbols. (In the figure above, $M = 2$, so each state may only emit an 'x' or a 'y'.) The emissions are also probabilistic: at each time step, the state i emits symbol k with probability β_{ik}, where β is an $N \times M$ matrix.

Thus what we observe is not a sequence of hidden states $Q = q_1\ q_2 \ldots\ q_t\ \ldots\ q_T$ but a sequence of emitted symbols $U = u_1\ u_2 \ldots\ u_t\ \ldots\ u_T$. (In our example, each q_t may take on the values 1, 2, 3 and each u_t may be an 'x' or a 'y'.)

Suppose we observe some sequence of emitted symbols U. If we happen to know the underlying hidden states Q, then the probability that the system assumes states Q and emits the symbols U is given by

$$\Pr(U,Q) = \Pr(q_1)\ \Pr(u_1|q_1)\ \Pr(q_2|q_1)\ \Pr(u_2|q_2) \ldots$$
$$= \pi(q_1)\ \beta(q_1,u_1)\ \alpha(q_1,q_2)\ \beta(q_2,u_2) \ldots$$

However, in the typical case, we do not actually know the sequence Q. In that case, one might try a brute force approach to computing $\Pr(U)$:

$$\Pr(U) = \sum_{\text{all } Q} \Pr(U, Q)$$

where the summation is done over all possible sequences of hidden states *Q*. This solution is rather inefficient and impractical for large *T*; fortunately an efficient solution (called the *forward procedure*) exists and is described in most introductory texts on Hidden Markov Models.

Now once again suppose we've observed a sequence *U*. Now we want to find the most probable sequence of hidden states *Q* that generated this give *U*. The problem is not well posed until we specify what the "most probable *Q*" means – are we maximizing the probability over the individual states q_t, over the entire sequence *Q*, etc. However, for most reasonable definitions of "most probable *Q*" there exist efficient ways of finding such a *Q*. In our approach we look for the *Q* that maximizes Pr(*U*,*Q*) over all sequences *Q*, and use the classic Viterbi algorithm to find it.

Finally, let us say we have observed a number of sequences *U*. We would like to estimate the likeliest α, β, and π that could generate these observations. This problem, known as *learning* or *training*, has no known global solution in closed form. We use the classic Baum-Welch expectation maximization algorithm to iteratively find an (α, β, π) at a local maximum of the function Pr(*U* | α, β, π).

6. Our Approach

Our objective was to demonstrate the utility of Hidden Markov Models in Hebrew vocalization. In order to do this, we compare three methods, to be described below in detail:

a. Context Free Dictionary Lookup
b. Dictionary Lookup with Parts of Speech
c. Dictionary Lookup with Hidden States

For each of these methods, we train on a corpus of pointed Hebrew text, and then check its performance on a novel test corpus.

A few remarks about the corpus are in order. We used the Hebrew Bible, vocalized and morphologically tagged by the Westminster Theological Seminary. For the figures presented here, we used to books of Genesis and Leviticus. The training corpus consisted of 1651 verses and contained 30743 words; the test corpus 166 verses and 2852 words. (The verses for the two corpora were randomly taken from Genesis and Leviticus.) The combined corpus contained 6562 distinct vocalized word types and 4762 distinct unvocalized word types. Articles, particles, and prepositions, which for the most part are attached to other words in Hebrew, were analyzed as separate words for this project.

We also comment that this was a difficult piece of training data. The grammar of the Bible is far from regular, the text contains numerous hapax legomena (words or expressions that only occur once in a corpus), and to complicate matters further, the pointing often varies with the chanting flags which were not included in this corpus. Training on modern prose would have undoubtedly yielded better results than those obtained with archaic verse.

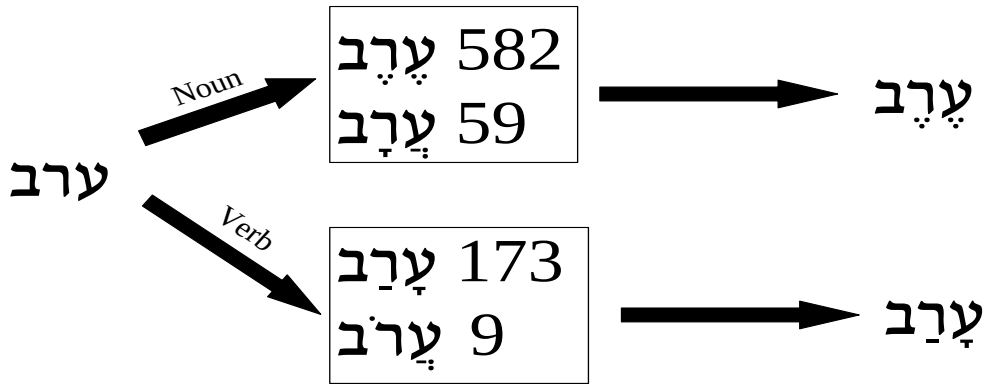
We proceed to discuss each of the three pointing methods.

a. Context Free Dictionary Lookup

[Figure 2. The most frequent pointing for the word ערב is used]

For each unpointed word in the corpus, its most frequent vocalization is used, where the frequencies are learned from the pointed corpus. No context is used, so for example the word ערב would always be pointed as עָרֶב.

b. Dictionary Lookup with Parts of Speech



[Figure 3. When ערב is a noun, it is pointed as עָרֶב; when it is a verb, it is pointed as עָרַב]

For this pointing method, a frequency table was created for each part of speech (there were 14 parts of speech used by the taggers). This method is not entirely context free – the way ערב gets vocalized depends on what part of speech it is[4].

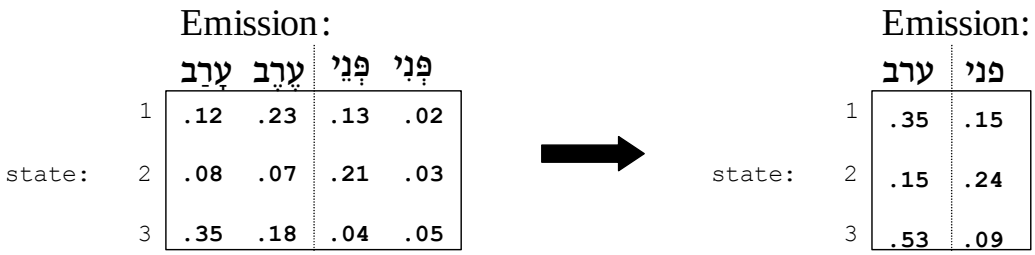
c. Dictionary Lookup with Hidden States

Before discussing the last method, let us provide the rationale behind it. We discovered that using the parts of speech tags does not significantly improve the accuracy of vocalization. Thus, rather than splitting the words into rigid part-of-speech categories, we decided to use an optimization algorithm (Baum-Welch) to find a more effective way to partition the words. The Westminster taggers used 14 part-of-speech tags, and so we arbitrarily decided to use an HMM with 14 hidden states.

[4] One might ask how the part-of-speech tags are assigned. For this project, we used the Westminster tags that came with the corpus. However, we had also built an HMM-based part of speech tagger that achieved an accuracy of over 97%.

Training

For the purposes of training, each pointed verse was treated as an observation *U*, where the *t*th word of the verse was u_t. Each pointed word was treated as a distinct emitted symbol. The algorithm then found a 14 × 14 matrix α and a 14 × 6562 matrix β that maximized to likelihood of the observed strings U[5]. The next step was to transform β into β' by the following "collapsing" operation:

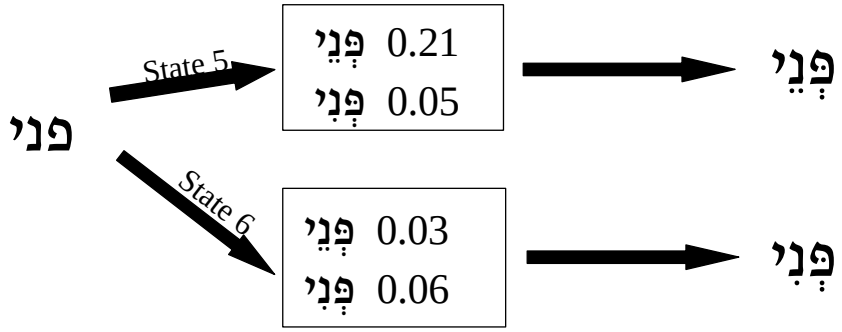


[Figure 4. For a given set of consonants, all the vocalized emission states are collapsed into a single unvocalized emission state, and the probabilities are summed across the state as shown.]

As illustrated in the figure, we partition the vocalized emitted symbols by consonants, where the words of every group have identical consonants but different pointings. For each such group in a given hidden state, we create an entry in β' in the same state-row, where the emission probability is the sum of the probabilities of the whole group.

Pointing

The pointing routine takes an unvocalized verse *U*, and (α,β') as inputs. It uses the Viterbi algorithm to find the most probable sequence of states *Q* underlying *U*. Having assigned a hidden state to each word, we proceed to choose the most frequent vocalization for that word according to its state (see figure).



[Figure 5. There is a separate emission probability table for each state, and so for example the word פני is vocalized as פְּנֵי when in state 5 but as פְּנִי when in state 6.]

7. Results

The results should be prefaced with the disclaimer that they were obtained on a very small training corpus and an even smaller test corpus, and are provided to illustrate general

[5] The reader might wonder what happened to the 14 × 1 matrix π. One of the states was constrained always to be the initial state, and so there was no need for a π matrix.

trends as opposed to statistically significant figures. Much more rigorous benchmarking will be performed on larger corpora of modern Hebrew (in later projects).
With this caveat in mind, we present the figures (where the accuracy is computed on a per-word basis):

- On the whole corpus (training + test)
  Context-free lookup: 74%
  Lookup with POS: 76%
  HMM Lookup: 83%

- On the test corpus
  Context-free lookup: 77%
  Lookup with POS: 79%
  HMM Lookup: 81%

An analysis of the β matrix after the training indicates that some of the states clearly preferred certain parts of speech (e.g., emitted verbal vocalizations with a high probability and adjectival ones with a low probability), while other states showed no part-of-speech preference, but might have been capturing other rules of Hebrew pointing.
The lookup with HMM has clearly outperformed the simple parts-of-speech lookup, while the parts-of-speech lookup does not do much better than context-free lookup. With only a few more parameters ($14 \times 14 = 196$), the training algorithm has captured the contextual dependencies better than the fixed part-of-speech tags. Though these preliminary results are more qualitative than quantitative, they indicate that even a crudely implemented Hidden Markov Model is a significant improvement over context-free lookup.

8. Future Goals

In the near future, I hope to build a robust morphological analyzer. The initial objective for this analyzer would be to provide an exhaustive list of possible vocalizations for a string of Hebrew consonants. For each vocalization, the engine would specify the derivation path, and ideally, would provide some probabilistic ranking of the vocalizations (i.e., most probable to least probable).
At first it seemed best to get in touch with the makers of *Nakdan* to obviate the need to start from scratch. However, *Nakdan*'s morphological analyzer has been shown here not to operate on general Hebrew principles, but rather to be largely based on lookup tables and ad-hoc rules. Therefore I have decided to start from scratch, guided by some current theoretical literature in the field.
I also have some ideas for combining the morphological analyzer with a Markov Model to capture some short-range contextual dependencies. One idea is to assign a Markov state to every possible form a Hebrew verb can take. Given a pointed verb, there is almost always a unique derivation for it. Thus the morphological analyzer could be used on pointed text to compute the Markov state of every verb. An analogous analysis can be performed on nouns and their attributes, as well as on other parts of speech. In my

estimation, a Markov Model could capture the contextual dependencies of the attributes with a high accuracy. To capture longer term dependencies, some rudimentary parsing could be performed. Perhaps a Hidden Markov Model could be used for capturing some semantic relations (the hidden states might be syntactic, while the observed states - semantic). These ideas are all very tentative and I plan to work on them during the next few years.
This ambitious project requires a large pointed corpus of modern Hebrew, which I am currently trying to obtain. I would also need dictionaries of nouns, verbs, adverbs, adjectives, foreign words etc, which I may need to build myself.

9. Acknowledgements

I would like to thank Chi-Lin Shih for suggesting the project, George Kiraz for consultations on computational Semitic linguistics, Moshe Dvir for the software consultations, Haim Sompolinksy for insightful comments, and most of all Daniel Lee for his thorough and patient guidance throughout the duration of this project.

10. References

Kiraz, George Anton. "Multi-Tiered Nonlinear Morphology Using Multi-Tape Finite Automata: A Case Study on Semitic." Paper in progress, 1999.

Manning, Christopher D. and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 1999.

Meron, Yoram. "Hebrew Text Analysis using Finite State Transducers." Lucent, Summer Project, 1998.

Nakdan Text. Commercial Software. HaMerkaz LeTechnologia Khinuchit, 1996.

Rabiner, Lawrence and Biing-Hwang Juang. *Fundamentals of Speech Recognition*. Prentice Hall, 1993.