Integrating Denotational Meaning into a DBN Language Model*

William Schuler, Tim Miller

Department of Computer Science and Engineering
University of Minnesota, Minnesota, U.S.A.
{schuler,tmill}@cs.umn.edu

Abstract

This paper describes a dynamic Bayes net (DBN) language model which allows recognition decisions to be conditioned on features of entities in some environment, to which hypothesized directives might refer. The accuracy of this model is then evaluated on spoken directives in various domains.

1. Introduction

The capacity to rapidly connect language to referential meaning is an essential aspect of human communication. Eye-tracking studies show that humans listening to spoken directives are able to actively attend to the entities that the words in these directives might refer to, even while these words are still being pronounced [1]. This timely access to referential (or *denotational*) information about input utterances may allow listeners to adjust their preferences among likely interpretations of noisy or ambiguous utterances to favor those that make sense in the current environment or discourse context, before any lower-level disambiguation decisions have been made.

If provided early enough in the recognition process, it is conceivable that this kind of information could significantly improve recognition accuracy of spoken language interfaces, particularly for applications in which users and interfaced systems have overlapping models of a common environment (e.g. event scheduling, navigating maps or 3-D simulations, setting up sensor networks, or directing robotic agents). Moreover, this immediate access to hypotheses about the referential meaning of input utterances could allow interfaced agents to provide incremental feedback to speakers using modalities other than speech – e.g. selecting or pointing at objects as they are described – so that speakers could adjust their descriptions of desired actions while they are still speaking, without having to wait for a post-recognition analysis of the complete utterance. Finally, a recognizer that estimates probabilities of input analyses based on the entities and relations they denote may be significantly easier to train and port across applications than one based only on word co-occurrences in text corpora, since the associations between words and entities that such a model requires in order to recognize input directives would be identical to those required to understand and execute these directives once they have been recognized. This re-use of training data could save considerable expense in applications where task requirements are relatively mutable and trained programmers are scarce.

This paper describes a probabilistic language model for a spoken language interface which bases recognition decisions on this kind of denotational meaning. The proposed language model resembles grammar-based or structural language models [2, 3], but instead of producing probability distributions on word strings or phrase structure trees, the model produces distributions on denotations (things in the world model that the utterance refers to), independent of the word string or tree used to convey this. The model does still calculate the set of possible word strings and trees used to convey denotations, but instead of finding the most probable string or tree (using Viterbi estimates), it marginalizes or sums them out (using forward probability estimates), leaving only a distribution on denoted entities. Other similar approaches [4, 5, 6] are either constrained to discrete environments such as databases, or are linguistically constrained to flat finite-state automata or incomplete context-free grammar parses and therefore not able to model a complete, recursive linguistic process from intended meaning to pronunciation, as the current approach attempts to do.

2. Formal framework

This model divides the top-down process of deriving an utterance from an intended denotation into three component probability models: one for semantic composition (choosing patterns of constituent structure with which to describe an intended denotation, based on a probabilistic context-free grammar); one for lexicalization (choosing words to describe an intended denotation); and one for attention (choosing other entities to use as landmarks in a description). This is done within a dynamic Bayes net (DBN) representation [7] corresponding to a variant of the Hierarchical Hidden Markov Model (HHMM) topology [8], which has been adapted to represent phrase structure trees with bounded center-recursion using a *right-corner transform*, which is the left-right dual of a left-corner transform [9]. This transforms all right-recursion in a context-free grammar into left-recursions of incomplete constituents (e.g. PP/NP for a prepositional phrase lacking a noun phrase yet to come).[1] A sample right-corner transform is shown in Figure 1. As with a left-corner transform, the right-corner transform minimizes the number of stack elements required to incrementally recognize a transformed grammar using a push-down automaton (PDA), so that the PDA stack will only be expanded in cases of center-recursion, which is notably rare in natural language.[2]

Each rule application in a thus-transformed syntactic corpus can then be used to calculate probability distributions for the memory elements in a finite stack of a pushdown automaton, based on the rightward depth of the node expanded by each

*This research is supported by grants from the Digital Technology Center Initiative Program and the Grant-In-Aid Program at the University of Minnesota Twin Cities, and by National Science Foundation CAREER award 0447685.

[1] The l/l' notation indicates an incomplete constituent with label l lacking a constituent with label l' to the right.

[2] As evinced by the difficulty humans show in recognizing heavily center-embedded but otherwise grammatical constructions, such as 'the cart [the horse [the man bought] pulled] broke.'

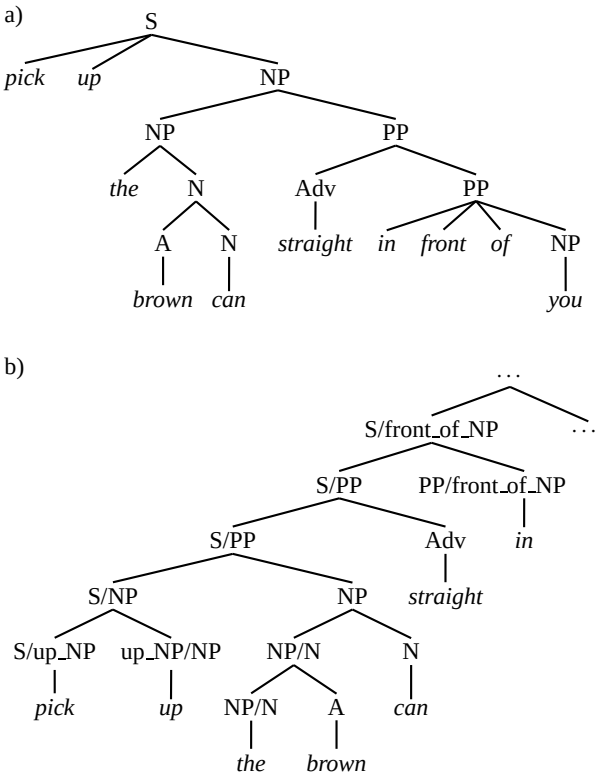

Figure 1: (a) ordinary phrase structure tree and (b) right-corner transform of this tree for the sentence 'pick up the brown can straight in front of you.'

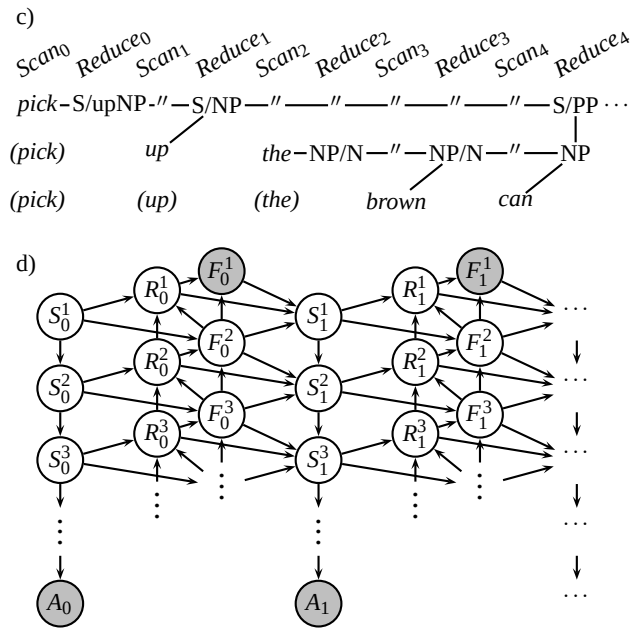

Figure 2: Right-corner derivation of 'pick up the brown can straight in front of you,' mapped to random variable positions in DBN (c), with dependencies as shown in (d). Taken together, each stack forms a complete analysis of the recognized input at every time frame t. Quote marks in (c) indicate labels copied from previous frame. Shaded nodes in (d) indicate observed evidence: A_t are observed frames of acoustical features, boolean F_t^1 are 'true' at the end of the utterance, 'false' otherwise.

rule application (see Figure 2). The model treats each memory element d in this finite stack, at each frame t of the input, as a separate random variable S_t^d with a distribution over labels $\langle c, \vec{e} \rangle$ (containing a category c and denoted entities $\vec{e}$) in a right-corner grammar. The model also introduces boolean random variables F_t^d, similar to those used in the Murphy-Paskin formulation of Hierarchic HMMs [8], in order to specify whether the HMM at depth d has reached a final state at frame t or not. Finally, the model introduces additional random variables R_t^d, in order to allow one or more constituents on the stack to be combined within a single frame (e.g. constituents 'NP/N' and 'brown' in frame 3, or constituents 'S/NP' and 'NP/N' and 'can' at frame 4 in Figure 2c), which is necessary in order to produce branching phrase structure derivations. This distinction has interesting correlation with the operations in an Earley parser for context-free grammars [10], with *reduce* operations taking place at R variables, and *predict* and *scan* operations taking place at S variables (depending on the values of the F variables, as in the Murphy-Paskin formulation). These S_t^d, R_t^d and F_t^d random variables can be repeated at each 10-millisecond speech frame, conditioned on the random variables for the previous frame and operations applied to the stack, in a layered Dynamic Bayes Net (DBN) representation [7]. This model is shown graphically in Figure 2d.

The independence assumptions in this model are:

$$\mathsf{P}(S_t^d \mid \text{all}) \doteq \mathsf{P}(S_t^d \mid F_{t-1}^d, F_{t-1}^{d+1}, S_t^{d-1}, R_t^d) \tag{1}$$

$$\mathsf{P}(R_t^d \mid \text{all}) \doteq \mathsf{P}(R_t^d \mid F_t^{d+1}, S_t^d, R_t^{d+1}) \tag{2}$$

$$\mathsf{P}(F_t^d \mid \text{all}) \doteq \mathsf{P}(F_t^d \mid F_t^{d+1}, S_t^{d-1}, R_t^d) \tag{3}$$

The definitions for S_t^d and R_t^d are then further broken down into 'composition' (Θ_C), 'attention' (Θ_A), and 'lexicalization' (Θ_L) components explained below, in which all f are true or false, all c are syntactic categories (e.g. 'S' or 'NP' or 'S/NP'), all $\vec{v}$ are entity coindexation or re-write patterns described below, and all $\vec{e}$ are tuples of entities from the environment that are referred to or *denoted* by this instance of category c (e.g. a single entity denoted by a noun phrase like 'the box', or a pair of related entities denoted by a preposition 'in'):

$$\begin{aligned}
&\mathsf{P}(S_t^d = \langle c, \vec{e} \rangle \mid F_{t\text{-}1}^d = f',\ F_{t\text{-}1}^{d+1} = f'',\ S_t^{d\text{-}1} = \langle c', \vec{e}' \rangle,\ R_t^d = \langle c'', \vec{e}'' \rangle) \\
&\quad = \mathsf{P}(\vec{v} \mid f',\ f'', c',\ \vec{e}',\ c'', \vec{e}'') \cdot \\
&\qquad \mathsf{P}(\vec{e} \mid \vec{v},\ f',\ f'', c',\ \vec{e}',\ c'', \vec{e}'') \cdot \\
&\qquad\quad \mathsf{P}(c \mid \vec{e},\ \vec{v},\ f',\ f'', c',\ \vec{e}',\ c'', \vec{e}'') \qquad (4) \\
&\quad \doteq \mathsf{P}_{\Theta_{C,d,f',f''}}(\vec{v} \mid c',\ c'') \cdot \\
&\qquad \mathsf{P}_{\Theta_A}(\vec{e} \mid \vec{v},\ \vec{e}',\ \vec{e}'') \cdot \\
&\qquad\quad \mathsf{P}_{\Theta_{L,d,f',f''}}(c \mid \vec{e},\ c',\ c'') \qquad (5)
\end{aligned}$$

The breakdown for R_t^d is essentially identical to that shown above for S_t^d except that it contains no term f''.

Each lexicalization model $\mathsf{P}_{\Theta_L}(c \mid \vec{e},\ c',\ c'')$ in the above equations is then calculated as the normalized product of the probability $\mathsf{P}_{\Theta_{L1}}(c \mid c',\ c'')$ of using category c in the context of categories c' and c'', times the probability $\mathsf{P}_{\Theta_{L2}}(\vec{x}_1..\vec{x}_a \mid c,\ c',\ c'')$ of describing each denoted entity e_i (with features $\vec{x}_i$) using cat-

egory c in context c', c'':

$$\mathsf{P}_{\Theta_L}(c \mid \vec{e}, c', c'') = \mathsf{P}_{\Theta_L}(c \mid e_1..e_a, c', c'') \quad (6)$$

$$\doteq \mathsf{P}_{\Theta_L}(c \mid \vec{x}_1..\vec{x}_a, c', c'') \quad (7)$$

$$= \frac{\mathsf{P}_{\Theta_{L1}}(c \mid c', c'') \cdot \mathsf{P}_{\Theta_{L2}}(\vec{x}_1..\vec{x}_a \mid c, c', c'')}{\sum_c \mathsf{P}_{\Theta_{L1}}(c \mid c', c'') \cdot \mathsf{P}_{\Theta_{L2}}(\vec{x}_1..\vec{x}_a \mid c, c', c'')} \quad (8)$$

Since models Θ_C and Θ_{L1} do not directly depend on entities, they can be extracted from phrase-structure- and reference-annotated training sentences collected in different environments from those used in evaluation. These are transcribed inputs from similar applications, whose phrasal and clausal constituents are enclosed with brackets and annotated with category labels (c) and vectors of denoted entities ($\vec{e}$). Training instances for $\mathsf{P}_{\Theta_C}(\vec{v} \mid c', c'')$ and $\mathsf{P}_{\Theta_{L1}}(c \mid c', c'')$ are then extracted from right-corner transformed and DBN-aligned versions of these sentences, with coindexation patterns $\vec{v}$ determined by the patterns of identical entities in the conditions and conclusions of these training instances.

The remaining models Θ_A and Θ_{L2} are directly based on entities, but can still be abstracted across environments using features of entities (e.g. relative position or size) rather the particular entities themselves. In the experiments described in Section 3, training instances for $\mathsf{P}_{\Theta_{L2}}(\vec{x}_1..\vec{x}_a \mid c, c', c'')$ were extracted from example images or feature vectors chosen by a trainer, and fit to Gaussian or other continuous distributions over entities, given a category context. In discrete environments, whose only features are boolean predicates over entities, these distributions were simply defined to be uniform over entities satisfying the appropriate predicate. The attention model $\mathsf{P}_{\Theta_A}(\vec{e} \mid \vec{v}, \vec{e}', \vec{e}'')$ was taken to be uniform for each 'NEW' entity in each experiment.

Viewed as a generative process, this language model begins with the composition model selecting a coindexation pattern $\vec{v}$ for a new constituent. This coindexation pattern contains an index pointer for each argument position i in $\vec{e}$, which points to the first entity in $\vec{e}$, $\vec{e}'$ or $\vec{e}''$ that exactly matches e_i, or is set to 'NEW' if position i contains the first occurrence of e_i. Once the coindexation pattern has been chosen, each 'NEW' entity is then selected from the environment using the attention model. For example, if a sentence constituent were being generated at the top level, a new entity would be chosen using a singleton coindexation of 'NEW'; or if some decomposition of a prepositional phrase were being generated, a 'NEW' landmark entity might be chosen for use in further description (as the NP complement of the PP), based on its proximity and relation to an existing (coindexed) trajector entity. Finally, the lexicalization model selects a word or syntactic category (or a multi-word/category combination such as 'in front of NP') to describe the chosen entity or relation among entities in a tuple.

By generating probabilities for hypotheses in this manner, the model can incrementally recognize right-corner derivations while still preserving explicit representations of intermediate constituents at all levels of the integrated model: e.g. representing subphone symbols[3] in the DBN's lowest ($d = 6$) level, partial phonemes in the next ($d = 5$) level,[4] partial words in the following ($d = 4$) level, and partial phrases and denotations at subsequent ($d \leq 3$) levels, until eventually the denotation of a complete sentence can be recognized in the top level, at the end of the utterance.

[3] These correspond to the onset, middle, and ending sounds of individual phonemes, whose distributions can be obtained using existing acoustical models.

[4] This level and the one below it are isomorphic to the state and emit variables in a Hidden Markov Model for subphone composition, which can also be extracted from existing acoustical models.

3. Evaluation

The accuracy of this denotational language model was evaluated in a live interface to a mobile robot, a (batch-mode) interface to a manipulator robot, and a (batch-mode) application for manipulating objects in a graphic display, each of which has a different kind of environment for calculating denotations.

3.1. Mobile robot interface

The robot interface used a relatively self-contained set of recognizable directives relating to wheel movement: '*start/stop moving*,' '*start/stop turning left/right*,' and the relevant sensors for the denotational recognizer: left and right wheel tachometers. 75 input utterances were collected by asking two subjects to direct a voice-controlled mobile robot using the above commands.

The Θ_{L2} component of the denotational language model (as described in Section 2) was trained on a small set of moving and turning scenarios staged by a trainer, consisting of three scenarios for each directive. These scenarios were intended to correspond to the preconditions of sample events that might be provided for each type of directive by an (experienced) user teaching the system different ways of changing trajectory. The Θ_C and Θ_{L1} components were trained on a hand-constructed annotated corpus listing the full set of possible directives.

The denotational model was compared with a baseline trigram Hidden Markov Model (HMM)-based language model trained on the sample set of directives described above. Of the 75 collected utterances, the integrated denotational model recognized 71 correctly, whereas the baseline HMM-based model recognized only 58 correctly. This represents a 70% reduction in recognition error due to the denotational model. This is a statistically significant improvement with $p \leq .01$ using a two-tailed t-test.

3.2. Manipulator robot interface

The denotational language model described above was also evaluated on collected directives to a voice-directed mobile manipulator arm in front of a shelf stacked with everyday household objects (cereal boxes, soft drink cans, etc.), which was photographed using a 3-D laser scanning camera.[5] The resulting 3-D point cloud was polygonized into a triangle mesh and segmented into entities e_i corresponding to convex regions of this mesh, each with continuous features $\vec{x}_{e_i}$ specifying the entity's size (exposed surface area), shape (ratio of longest to second longest perpendicular dimensions), spatial location (3-D coordinates of centroid), and color (average hue, saturation, and intensity over all pixels in the segment). Word meanings in Θ_{L2} were modeled for adjectives and prepositions using multivariate Gaussians in this feature space (defined on color, size, and shape features for adjectives, and on differences in centroid coordinates for prepositions), which were developed partially by hand as a domain-independent language resource. Verbs and common nouns were considered domain-specific and were trained

[5] Subjects were asked to direct the manipulator arm to pick up several objects from the shelf. The objects were visually designated (by pointing), in order to avoid biasing subjects toward any linguistic description. As a result, some of the collected directives contain very long, complex definite descriptions. The manipulator arm was a non-functional prop during this data collection.

automatically on a version of the collected corpus of arm directives that was annotated with phrase structure (labeled brackets) and constituent denotations (in the associated training environment). The Θ_C and Θ_{L1} components were trained on (right-corner transforms of) the denotation-annotated phrase structure trees in this same annotated corpus. All training and testing using this corpus was done using the leave-one-out method of cross-validation.

The accuracy of the sentence-level denotations obtained from the integrated denotational language model was tested against that of denotations obtained through the widely-used practice of parsing and interpreting the single sentence output of a trigram HMM language model, trained on transcriptions of the same collected corpus, using a parser and interpreter trained on the annotated version of the same corpus (again using leave-one-out cross-validation). However, primarily due to a high rate of speech repairs for this somewhat unnatural task of directing a manipulator arm in a unimodal interface (i.e. without pointing), the HMM language model yielded a relatively high (47%) word error rate on the collected utterances,[6] with no sentences yielding correct denotations. The denotational model did yield 54 parses (of 165 total), but only 20% of these had correct denotations ($p<.1$ due to chance) – a nevertheless statistically significant improvement ($p<.01$ using a two-tailed t-test), which was evenly distributed across task environments.

3.3. Graphical display interface

The final evaluation of the denotational language model was performed in a discrete environment, within an application for manipulating 2-D objects on a graphical display. The graphical display consisted of nested colored boxes, some of which were linked to other boxes using thick black lines. The task was intended to represent a directory structure in a form that could be instantly perceived by test subjects. The sentences were roughly similar to those collected in the manipulator arm environment. Again, an annotated corpus of 160 collected utterances was used for training Θ_C and Θ_{L1} (with leave-one-out cross-validation in testing), but the discrete relations were handled using a uniform distribution over satisfying entities in Θ_{L2}. Since this task was somewhat more natural than the manipulator arm task, the HMM had a word error rate of only 20%, but still did not produce any entirely correct sentences which could be successfully parsed and interpreted using a parser trained on transcriptions of these utterances (largely because it most often missed short but semantically significant words like 'in'), whereas the denotational model was able to parse and correctly recognize denoted entities in approx. 50% of inputs (approx. 10 candidate entities per environment; improvement $p<.01$).

4. Conclusions and future work

This paper has described a DBN language model which allows recognition decisions to be influenced by information about the entities or relations (tuples of entities) denoted by spoken directives. Unlike similar approaches, the model provides complete, recursive, probabilistic analyses of language production (with unlimited left- or right-recursion and human-like memory limits on internal recursion in syntax). This model has been shown to provide more accurate recognition than conventional HMM-based language models. In addition, this denotational model has two interesting properties not found in other models:

1. Since it incrementally models the denotational meaning of each utterance, it can be configured to provide incremental feedback at a referential level (using other modalities such as gesture, gaze, or graphical displays to indicated hypothesized denotations); and
2. Since it incrementally models the denotational meaning of each utterance, it does not need to compute (e.g. Viterbi) most likely sequences of words. This means it can be configured to run as a continuous recognition process, providing overlapping feedback in other modalities without having to wait for utterance boundaries.

These possibilities will be explored in future work in the context of a general multi-sensor interface architecture for communicative agents. Further information about this research can be found at `http://www.cs.umn.edu/research/nlp`.

[6] Average sentence length was >19 words.

5. References

[1] M. K. Tanenhaus, M. J. Spivey-Knowlton, K. M. Eberhard, and J. E. Sedivy, "Integration of visual and linguistic information in spoken language comprehension," *Science*, vol. 268, pp. 1632–1634, 1995.

[2] C. Chelba and F. Jelinek, "Exploiting syntactic structure for language modeling," in *Proc. COLING/ACL '98*, Montreal, Canada, 1998, pp. 225–231.

[3] E. Charniak, "Immediate-head parsing for language models," in *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, Toulouse, France, 2001, pp. 116–123.

[4] N. Haddock, "Computational models of incremental semantic interpretation," *Language and Congnitive Processes*, vol. 4, pp. 337–368, 1989.

[5] D. Roy, P. Gorniak, N. Mukherjee, and J. Juster, "A trainable spoken language understanding system for visual object selection," in *Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP'02)*, 2002, pp. 593–596.

[6] P. Gorniak and D. Roy, "Grounded semantic composition for visual scenes," *Journal of Artificial Intelligence Research*, vol. 21, pp. 429–470, 2004.

[7] T. Dean and K. Kanazawa, "A model for reasoning about persitence and causation," *Computational Intelligence*, vol. 5, no. 3, pp. 142–150, 1989.

[8] K. P. Murphy and M. A. Paskin, "Linear time inference in hierarchical HMMs," in *Proceedings of Neural Information Processing Systems*, 2001, pp. 833–840.

[9] S. J. Rosenkrantz and P. M. Lewis, II, "Deterministic left corner parser," in *IEEE Conference Record of the 11th Annual Symposium on Switching and Automata*, 1970, pp. 139–152.

[10] J. Earley, "An efficient context-free parsing algorithm," *CACM*, vol. 13, no. 2, pp. 94–102, 1970.