SEMANTIC MAPPING OF NATURAL LANGUAGE INPUT TO DATABASE ENTRIES VIA CONVOLUTIONAL NEURAL NETWORKS

Mandy Korpusik, Zachary Collins, and James Glass

MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, MA 02139, USA
{korpusik, zcollins, glass}@mit.edu

ABSTRACT

Natural language processing research has made major advances with the concept of representing words, sentences, paragraphs, and even documents by embedded vector representations. We apply this idea to the problem of relating foods, as expressed in natural language meal descriptions, to corresponding database entries. We generate fixed-length embeddings for U.S. Department of Agriculture (USDA) food database entries, as well as vector-based representations of natural language meal descriptions, through a convolutional neural network (CNN) architecture that predicts whether or not a USDA food item is present in the meal description. We compute dot products between each token in a meal description and a USDA food entry. By ranking the network's predicted average dot product between each possible database food entry and a meal description, we show it is possible to directly predict the USDA foods mentioned in a meal without requiring intermediate steps that would be used in a conventional database access application. We report the performance of this model on a binary verification task of over 48k meal descriptions, and show that this approach, when integrated with a Markov model, substantially outperforms our previous best multi-stage approach involving a conditional random field tagger, probabilistic segmentation, and database lookup.

Index Terms— Convolutional Neural Networks, Finite State Transducers, Crowdsourcing

1. INTRODUCTION

Many speech understanding applications involve mapping the concepts present in a natural language expression to their corresponding database entries. In our recent efforts to create an ability for users to log their food intake by speech, we are faced with this situation, where we need to determine which particular foods have been described, and, ultimately, find their associated database entry in a nutritional database. For example, if someone says, "For breakfast I had a bowl of Kellogg's corn flakes," we would like to find the matching food entries in the USDA nutrient database (i.e., "Cereals ready-to-eat, KELLOGG, KELLOGG'S Corn Flakes"), so we can log the appropriate nutrition information.

As illustrated in Figure 1, our initial work on this problem [1, 2] used conventional statistical approaches, whereby we fed meal descriptions through a pipeline involving 1) semantic tagging of each token using a conditional random field (CRF) model, 2) performing a kind of food segmentation by using classifiers to associate foods with their associated properties (e.g., brands, quantities etc), and 3) retrieving matching USDA food entries from a database. This approach had several undesirable properties. First, it was vulnerable to intermediate errors. Second, it relied on heuristics for the database lookup (e.g., stemming) to account for mismatches in language usage between the meal description and the database entry. Unfortunately, there are many instances where the common word (e.g., "toast") is not present in the database entry at all, so additional heuristics are required to bridge this mismatch [3]. Finally, our data collection efforts focused on collecting natural language descriptions and their associated semantic annotations, so we could not accurately quantify the overall performance of the system.

To address these issues, we have developed a model that can directly map between the natural language description to the underlying database entries. In the following sections we describe our data collection efforts, the new CNN-based model that ranks database entries matching a food description, and experiments that show that this model substantially outperforms our previous best configuration.

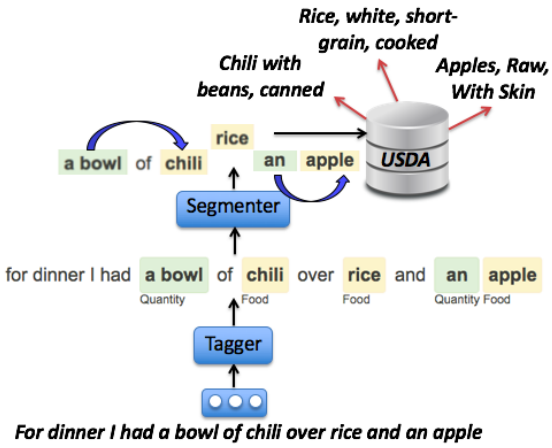

Fig. 1. The previous system's sequence of steps: CRF tagging, segmenting a meal into food entities, and USDA database lookup.

2. DATA COLLECTION

Previously [1], we collected 22k meal descriptions and associated semantic annotations via crowd-sourcing with Amazon Mechanical Turk. However, there were problems with these data: we did not know the correct USDA answers for each meal description. In this work, we adopt a different strategy by having Turkers generate a meal description that matches a selected subset of USDA items. This enables us to build models that directly map from meal descriptions

This research was sponsored by a grant from Quanta Computing, Inc., and by the Department of Defense (DoD) through the National Defense Science Engineering Graduate Fellowship (NDSEG) Program.

to USDA foods. We do not know the order or location of the foods in the meal description, but these can be inferred automatically.

In order to generate "reasonable" meal description tasks, we partitioned the over 5k foods in the USDA database into specific meals such as breakfast, dinner, etc. (see Table 1). We also had a special set of 101 food items that we are using for a pilot user study we are conducting with nutritionists from Tufts University. A given task was randomly assigned a subset of 9-12 food items from different categories. To reduce biasing the language used by Turkers, we included images of the food items along with the less natural USDA titles. Turkers were asked to select at least three of the foods, and generate a meal description using these items. This enabled workers to select foods that would typically be eaten together, producing more natural meal descriptions and quantities.

Meal	# Foods	# Diaries	# Words per Diary
Breakfast	1167	4010	18.8
Dinner	2570	3850	21.6
Salad	232	4040	19.1
Sandwiches	375	4000	20.1
Smoothies	384	3850	20.1
Pasta/Rice	1270	4000	20.6
Snacks	1342	4077	19.1
Fast Foods/Meals	669	3886	19.1
All Foods	5124	31712	19.8
101 Foods	101	16589	18.2

Table 1. Meal description statistics, organized by category.

3. METHOD

We employed two steps to achieve a system that directly selects the best USDA matches for a given meal description: 1) we constructed a CNN model that learns vector representations for USDA food items through a binary verification task (i.e., whether or not a USDA item is mentioned in a meal description), and 2) we incorporated the model predictions into a Markov model framework to select the most likely set of USDA foods in the meal.

3.1. Convolutional Neural Network Model

As shown in Figure 2, the model is composed of two CNNs: one for the meal description and one for a USDA food. The text is first tokenized using spaCy (https://spacy.io). Each CNN contains a 50-dimension embedding layer and a 1D convolution of 64 filters spanning a window of three tokens with a rectified linear unit (ReLU) activation. The USDA component follows the CNN with a max-pooling layer over the input tokens to produce a 64 dimensional vector representation. During training this is followed by a dropout of probability 0.2 (i.e., randomly set 20% of units to 0 at each update during training), and batch normalization to maintain a mean near zero and a standard deviation close to one.

The meal description component similarly uses dropout and normalization during training. Following this, a dot product is performed with the USDA vector and each 64 dimensional CNN output of the meal description (i.e., with each token). Mean-pooling is then performed across these dot products to produce a single scalar value, which we force to be between zero and one with a sigmoid layer.

To prepare the data for training, we padded each USDA food entry to 20 tokens and each meal description to 100 tokens, and we limited the vocabulary to the most frequent 3,000 words. Our attempts

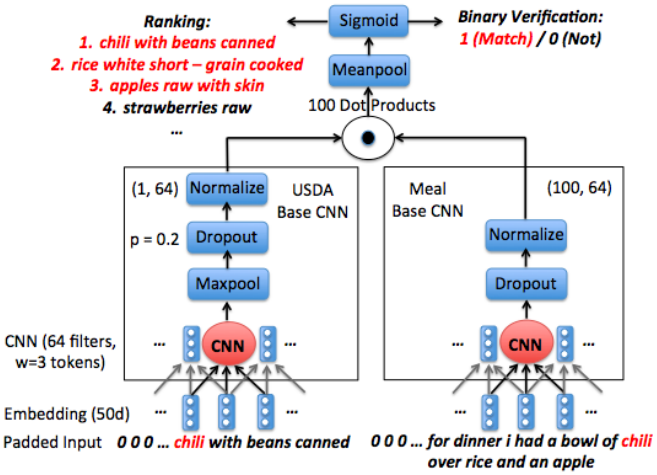

Fig. 2. Architecture of our CNN model for predicting whether a USDA food entry is mentioned in a meal description.

to remove bias due to the padding by masking the dot products only seemed to hurt performance.

We trained the model to predict each (USDA food, meal) input pair as a match or not (i.e., 1 or 0) with a threshold of 0.5 on the output. The model learns with the Adam optimizer [4] on binary cross-entropy loss, norm clipping at 0.1, a learning rate of 0.001, early stopping on the validation data (i.e., 20% of the data), and mini-batches of 16 samples. We removed all capitalization and removed commas from the USDA food entry names. For each positive USDA hit, we randomly selected a negative sample from the USDA items not mentioned in the meal.

3.2. Sequential Markov Model

While the CNN model learns to predict whether or not a USDA food is mentioned in a meal, it does not directly retrieve the matching USDA foods from a meal. To do this, we use a Markov model framework to take as input the token-by-token dot products (between each token in a meal and all possible USDA foods) and output the most likely sequence of USDA foods in the meal. The token-by-token dot products for each meal description are represented by a meal Finite-State Transducer (FST), M, with each state corresponding to a particular token/USDA food match with weights corresponding to the negative dot products for each token with each USDA food entry.

To enable decoding of the n-best token alignments, we construct a food FST, F, that represents a generic food Markov model whose states consist of USDA foods or an "Other" state (Fig. 3). The model we used in these current experiments enables transitions between "Other" states (i.e., no USDA food is discussed at that token) and three interleaved USDA food states, where there are transitions for each possible USDA entry. Each state has a self-loop, which allows multiple consecutive tokens to have the same USDA food. To decode a particular meal description, we compose M with F and compute the top-5 sequences of USDA foods with n-best Viterbi decoding.

3.2.1. Training Alignments

Composing the meal FST M with the food FST F yields token-level alignments. For training, we can use a constrained meal FST that only considers the known USDA foods in the particular meal

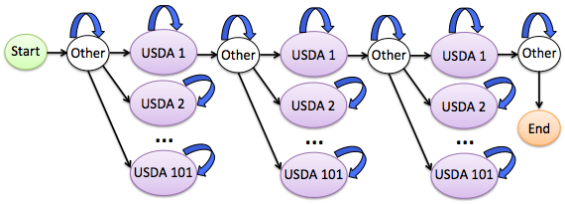

Fig. 3. A food FST, F, for predicting the USDA food sequence in a meal given the NN's predicted dot products. Restricting meals to three foods is a limitation, so we will increase flexibility in the future.

descriptions. We also use the resulting alignments to generate more specific positive/negative training examples to improve the CNN performance. For each input (USDA food, meal) pair, we generate a positive example of a shortened food segment and its corresponding USDA entry (and a negative example with an incorrect USDA entry), where the segment is all the tokens that were aligned with that USDA food in the alignment. Thus, we fine-tune the network so that it knows which tokens are associated with which USDA entries.

3.2.2. Segmentation

At test time, we use the n-best results for segmenting meal descriptions into food entities and predicting alternative USDA items for each entity. We are not interested in minor token-level re-alignments however, and just care about unique USDA item sequences. To force the n-best decoder to generate different USDA food choices, we compose the meal and food FSTs with a third high-level FST that outputs nothing for "Other" states and self-loops. From the positions of the USDA foods in this final sequence of IDs, we can determine which food alternatives group together and their ranking.

4. EXPERIMENTS

To train our models and tune hyperparameters, we split our data into validation (20%) and training data (80%). A held-out test set was used for evaluation. We report accuracy on the verification task, mean average precision (MAP) scores for ranking (i.e. ranking USDA foods in a meal), and a comparison of this approach to the previous setup of tagging, segmenting, and database lookup.

4.1. Binary Verification Task

For each meal category, we report the training and validation accuracy in Table 2 on the binary verification task (i.e., predicting whether a USDA food item is mentioned in a meal description). We can see that performance is higher for meals that have fewer foods (e.g., salads) and is lower for more complex meals such as dinner.

4.2. 101 Foods Case Study

For all subsequent experiments, we focused on a subset of 101 USDA foods (i.e., 16,589 total meal descriptions) for training and evaluation, which we pay special attention to since these are the foods we will use in a pilot study. In Table 3, we show the performance of the CNN model for binary verification and ranking (i.e., how highly it ranks the correct USDA foods in a meal description). We measure the ranking performance, which is more closely aligned to what the system must accomplish at test time, using mean average precision (MAP).

Meal	Train Acc.	Val. Acc.
Breakfast	87.5	81.3
Dinner	87.3	79.8
Salad	89.5	87.6
Sandwiches	85.9	82.2
Smoothies	89.5	85.5
Pasta/Rice	88.1	84.0
Snacks	85.6	78.5
Fast Foods/Meals	90.2	83.4

Table 2. Binary verification scores for each meal category.

$$MAP = \frac{\sum_{k=1}^{n} prec(k) rel(k)}{\text{correct \# foods in the meal}} \quad (1)$$

where k is the rank of a USDA food entry, $prec(k)$ is the precision of the food entry at rank k (i.e., how many correct foods have been identified so far, divided by the current rank k), and $rel(k)$ is 1 if the USDA food at rank k is in the meal description, and 0 otherwise.

We previously compared the LSTM to the CNN for semantic tagging [5], but in this work we chose the CNN for our experiments because it has comparable performance to the LSTM while using fewer parameters and training faster. We also observe that incorporating aligned token segments specific to each USDA food as additional training examples improved performance.

Model	Train Acc.	Val. Acc.	MAP
baseline word matcher	59.6	59.1	0.067
baseline classifier	92.2	90.8	0.076
CNN	96.0	94.5	0.889
CNN + alignments	**97.2**	**96.1**	**0.907**

Table 3. Performance on 101 foods. The baselines are a lexical matcher (i.e., at least two shared words is a match) and logistic regression classifier trained on n-grams, word match counts, and learned vector dot products.

4.3. System Evaluation

We ultimately care how well the system will perform at test time with actual users, so we also compared the performance of the existing system set up with tagging followed by food-property association and rule-based database lookup to the performance using the new neural architecture followed by the FST n-best prediction. We evaluated the system by the percent of correct USDA entries recalled. As shown in Table 4, the baseline CRF tagger with many manually defined features (including n-grams, part-of-speech tags, word vectors and prototype similarity, clusters, and capitalization, etc.) performs similarly to the CNN tagger with no feature engineering, when connected with the conventional SQL-based lookup in the USDA database (Figure 1). In contrast, the CNN-produced food embedding vector combined with FST n-best decoding has a much higher recall than either tagger, from 79% up to 92.6%. Note that although the FST decoding framework naturally lends itself to generating alternative hypotheses for different values of n, for purposes of this evaluation, we only report top-5 recall.

Model	USDA Recall
Baseline CRF tagging with SQL lookup	78.9%
CNN tagging with SQL lookup	78.5%
CNN food embeddings with FST 5-best	**92.6%**

Table 4. System evaluation with tagging followed by USDA lookup vs. neural-FST setup with direct USDA predictions on a held-out test set of 5k meals with only 101 USDA foods.

4.4. Analysis

In this section, we show through qualitative analysis that the neural network (NN) model is indeed learning meaningful vector representations of the USDA food entries, which is why it performs so well on ranking the matching USDA foods in a meal description.

If we look at the nearest neighbor to three USDA foods (see Table 5) using Euclidean distance, we observe that the neighbors are semantically similar. Looking at the NN model's predicted dot products between USDA foods and each token in a meal description, we observe spikes at tokens corresponding to that USDA food entry. We visualize the spike profile of the dot products at each token for the top USDA hits in Fig. 4. The USDA foods "McDonald's Big Mac" and "Fast Foods, Cheeseburger" spike at "big mac," whereas "Catsup" peaks at "ketchup," and "Bananas, Raw" at "peeled banana."

USDA Food	Nearest USDA Food
Rice white short-grain...	...Mexican Spanish Rice
Fast Foods Chicken Tenders	Chicken Broiler or Fryers...
Beans Baked Canned...	Beans Black Mature...

Table 5. Nearest 101 foods to three USDA learned food vectors.

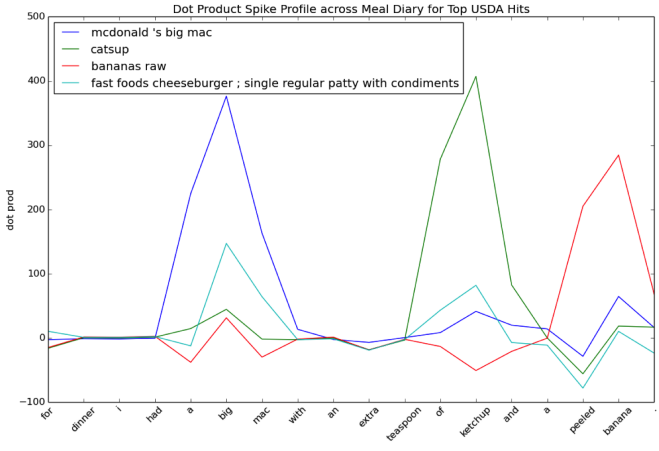

Fig. 4. Dot products between top USDA hits and meal tokens for the meal description *"for dinner I had a big mac with an extra teaspoon of ketchup and a peeled banana."*

5. RELATED WORK

While in our work we learn embeddings for USDA food entries through CNNs, recent work [6] has analyzed the relative strengths of various other sentence embeddings, including averaging word vectors learned with the continuous-bag-of-words method [7], LSTM auto-encoders [8], and skip-thought vectors based on gated recurrent units (GRU) [9]. Our approach differs from these in that we use a CNN rather than recurrent networks, and we learn the vectors through a domain-specific task for predicting whether a USDA food entry matches a meal description.

Similar work in learning joint embeddings for two different modalities or languages have explored a margin-based contrastive loss, which would be interesting to compare against our binary verification cross-entropy loss. For ranking annotations given an image, prior work directly incorporated the rank into the model's loss function, along with a hinge loss between true and false annotation samples [10]; similarly, a margin-based loss was used to learn a joint multimodal space between images and captions for caption generation [11, 12], and sentence/document embeddings were learned through a multilingual parallel corpus with a noise-contrastive hinge loss ensuring non-aligned sentences were a certain margin apart [13]. Other related work predicted the most relevant document given a query through the cosine similarity of jointly learned embeddings based on bag-of-words term frequencies [14].

Many researchers are now exploring CNNs for natural language processing (NLP). For example, in question answering, recent work has shown improvements using deep CNN models for text classification [15, 16, 17] following the success of deep CNNs for computer vision. Whereas these architectures take in a simple input text example and predict a classification label, our task takes in two input sentences and predicts whether they match. In work more similar to ours, parallel CNNs predict the similarity of two input sentences. While we process each input with its own CNN, others first compute a word similarity matrix between the two sentences (like an image matrix of pixels) and use the matrix as input to one CNN [18, 19, 20].

Attention-based CNN (ABCNN) models have also been proposed for sentence matching. The ABCNN [21] combines two approaches: applying attention weights to the input representations before convolution, as well as after convolution but before pooling. Our method is similar, but we compute dot products (our version of the attention scheme) with the max-pooled high-level representation of the USDA vector. Hierarchical ABCNN applies cosine similarity attention between CNN representations of a query and each sentence in a document for machine comprehension [22]. Thus, the attention comes after pooling across the input, whereas we compute the dot products between each meal token and the learned USDA vector.

6. CONCLUSION AND FUTURE WORK

In this paper, we have demonstrated a novel neural network-based technique with FST decoding for learning a food embedding space that enables directly translating natural language meal descriptions into their corresponding USDA food database entries, without requiring any semantic tagging intermediate steps and avoiding database lookup heuristics. We achieve 92.6% recall on a held-out set of 5,000 meal descriptions containing a limited set of 101 USDA food options, whereas our previous baseline using a CRF tagger combined with database lookup only achieves 79% recall. To extend our work, we plan to collect more data for the other USDA foods, not just the limited set of 101 food entries, which will enable us to expand the coverage of the new approach to all possible USDA foods. In addition, since users often write typos or misspellings in their meal descriptions, we would like to incorporate character-based embeddings [23, 24, 16], rather than only learning word embeddings.

7. REFERENCES

[1] M. Korpusik, N. Schmidt, J. Drexler, S. Cyphers, and J. Glass, "Data collection and language understanding of food descriptions," *Proc. SLT*, 2014.

[2] M. Korpusik, C. Huang, M. Price, and J. Glass, "Distributional semantics for understanding spoken meal descriptions," *Proc. ICASSP*, 2016.

[3] R. Naphtal, "Natural language processing based nutritional application," M.S. thesis, Massachusetts Institute of Technology, 2015.

[4] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[5] M. Korpusik and J. Glass, "Spoken language understanding in a nutrition dialogue system," *ASLP*, Submitted.

[6] Y. Adi, E. Kermany, Y. Belinkov, O. Lavi, and Y. Goldberg, "Fine-grained analysis of sentence embeddings using auxiliary prediction tasks," *arXiv preprint arXiv:1608.04207*, 2016.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[8] J. Li, M. Luong, and D. Jurafsky, "A hierarchical neural autoencoder for paragraphs and documents," *arXiv preprint arXiv:1506.01057*, 2015.

[9] R. Kiros, Y. Zhu, R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, "Skip-thought vectors," in *Proc. NIPS*, 2015, pp. 3294–3302.

[10] J. Weston, S. Bengio, and N. Usunier, "Wsabie: Scaling up to large vocabulary image annotation," 2011.

[11] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3128–3137.

[12] D. Harwath, A. Torralba, and J. Glass, "Unsupervised learning of spoken language with visual context," in *Proc. NIPS*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds., pp. 1858–1866. Curran Associates, Inc., 2016.

[13] K. Hermann and P. Blunsom, "Multilingual models for compositional distributed semantics," *arXiv preprint arXiv:1404.4641*, 2014.

[14] P. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2013, pp. 2333–2338.

[15] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for natural language processing," *arXiv preprint arXiv:1606.01781*, 2016.

[16] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. NIPS*, 2015, pp. 649–657.

[17] Y. Xiao and K. Cho, "Efficient character-level document classification by combining convolution and recurrent layers," *arXiv preprint arXiv:1602.00367*, 2016.

[18] L. Pang, Y. Lan, J. Guo, J. Xu, S. Wan, and X. Cheng, "Text matching as image recognition," *arXiv preprint arXiv:1602.06359*, 2016.

[19] Z. Wang, H. Mi, and A. Ittycheriah, "Sentence similarity learning by lexical decomposition and composition," *arXiv preprint arXiv:1602.07019*, 2016.

[20] B. Hu, Z. Lu, H. Li, and Q. Chen, "Convolutional neural network architectures for matching natural language sentences," in *Proc. NIPS*, 2014, pp. 2042–2050.

[21] W. Yin, H. Schütze, B. Xiang, and B. Zhou, "Abcnn: Attention-based convolutional neural network for modeling sentence pairs," *arXiv preprint arXiv:1512.05193*, 2015.

[22] W. Yin, S. Ebert, and H. Schütze, "Attention-based convolutional neural network for machine comprehension," *arXiv preprint arXiv:1602.04341*, 2016.

[23] Y. Kim, Y. Jernite, D. Sontag, and A. Rush, "Character-aware neural language models," *arXiv preprint arXiv:1508.06615*, 2015.

[24] C. Santos and B. Zadrozny, "Learning character-level representations for part-of-speech tagging," in *Proc. ICML*, 2014, pp. 1818–1826.