A Neural Network for Factoid Question Answering over Paragraphs

Mohit Iyyer[1], **Jordan Boyd-Graber**[2], **Leonardo Claudino**[1], **Richard Socher**[3], **Hal Daumé III**[1]
[1]University of Maryland, Department of Computer Science and UMIACS
[2]University of Colorado, Department of Computer Science
[3]Stanford University, Department of Computer Science
{miyyer,claudino,hal}@umiacs.umd.edu,
Jordan.Boyd.Graber@colorado.edu, richard@socher.org

Abstract

Text classification methods for tasks like factoid question answering typically use manually defined string matching rules or bag of words representations. These methods are ineffective when question text contains very few individual words (e.g., named entities) that are indicative of the answer. We introduce a recursive neural network (RNN) model that can reason over such input by modeling textual compositionality. We apply our model, QANTA, to a dataset of questions from a trivia competition called quiz bowl. Unlike previous RNN models, QANTA learns word and phrase-level representations that combine across sentences to reason about entities. The model outperforms multiple baselines and, when combined with information retrieval methods, rivals the best human players.

1 Introduction

Deep neural networks have seen widespread use in natural language processing tasks such as parsing, language modeling, and sentiment analysis (Bengio et al., 2003; Socher et al., 2013a; Socher et al., 2013c). The vector spaces learned by these models cluster words and phrases together based on similarity. For example, a neural network trained for a sentiment analysis task such as restaurant review classification might learn that "tasty" and "delicious" should have similar representations since they are synonymous adjectives.

These models have so far only seen success in a limited range of text-based prediction tasks, where inputs are typically a single sentence and outputs are either continuous or a limited discrete set. Neural networks have not yet shown to be useful for tasks that require mapping paragraph-length inputs to rich output spaces.

Later in its existence, this polity's leader was chosen by a group that included three bishops and six laymen, up from the seven who traditionally made the decision. Free imperial cities in this polity included Basel and Speyer. Dissolved in 1806, its key events included the Investiture Controversy and the Golden Bull of 1356. Led by Charles V, Frederick Barbarossa, and Otto I, for 10 points, name this polity, which ruled most of what is now Germany through the Middle Ages and rarely ruled its titular city.

Figure 1: An example quiz bowl question about the Holy Roman Empire. The first sentence contains no words or named entities that by themselves are indicative of the answer, while subsequent sentences contain more and more obvious clues.

Consider factoid question answering: given a description of an entity, identify the person, place, or thing discussed. We describe a task with high-quality mappings from natural language text to entities in Section 2. This task—quiz bowl—is a challenging natural language problem with large amounts of diverse and compositional data.

To answer quiz bowl questions, we develop a dependency tree recursive neural network in Section 3 and extend it to combine predictions across sentences to produce a question answering neural network with trans-sentential averaging (QANTA). We evaluate our model against strong computer and human baselines in Section 4 and conclude by examining the latent space and model mistakes.

2 Matching Text to Entities: Quiz Bowl

Every weekend, hundreds of high school and college students play a game where they map raw text to well-known entities. This is a trivia competition called *quiz bowl*. Quiz bowl questions consist of four to six sentences and are associated with factoid answers (e.g., history questions ask players to identify specific battles, presidents, or events). Every sentence in a quiz bowl question is guaranteed to contain clues that uniquely identify its answer, even without the context of previous sentences. Players answer at any time—ideally more quickly than the opponent—and are rewarded for correct answers.

Automatic approaches to quiz bowl based on existing NLP techniques are doomed to failure. Quiz bowl questions have a property called *pyramidality*, which means that sentences early in a question contain harder, more obscure clues, while later sentences are "giveaways". This design rewards players with deep knowledge of a particular subject and thwarts bag of words methods. Sometimes the first sentence contains no named entities—answering the question correctly requires an actual understanding of the sentence (Figure 1). Later sentences, however, progressively reveal more well-known and uniquely identifying terms.

Previous work answers quiz bowl questions using a bag of words (naïve Bayes) approach (Boyd-Graber et al., 2012). These models fail on sentences like the first one in Figure 1, a typical hard, initial clue. Recursive neural networks (RNNs), in contrast to simpler models, can capture the compositional aspect of such sentences (Hermann et al., 2013).

RNNs require many redundant training examples to learn meaningful representations, which in the quiz bowl setting means we need multiple questions about the same answer. Fortunately, hundreds of questions are produced during the school year for quiz bowl competitions, yielding many different examples of questions asking about any entity of note (see Section 4.1 for more details). Thus, we have built-in redundancy (the number of "askable" entities is limited), but also built-in diversity, as difficult clues cannot appear in every question without becoming well-known.

3 Dependency-Tree Recursive Neural Networks

To compute distributed representations for the individual sentences within quiz bowl questions, we use a dependency-tree RNN (DT-RNN). These representations are then aggregated and fed into a multinomial logistic regression classifier, where class labels are the answers associated with each question instance.

In previous work, Socher et al. (2014) use DT-RNNs to map text descriptions to images. DT-RNNs are robust to similar sentences with slightly different syntax, which is ideal for our problem since answers are often described by many sentences that are similar in meaning but different in structure. Our model improves upon the existing DT-RNN model by jointly learning answer and question representations in the same vector space rather than learning them separately.

3.1 Model Description

As in other RNN models, we begin by associating each word w in our vocabulary with a vector representation $x_w \in \mathbb{R}^d$. These vectors are stored as the columns of a $d \times V$ dimensional word embedding matrix W_e, where V is the size of the vocabulary. Our model takes dependency parse trees of question sentences (De Marneffe et al., 2006) and their corresponding answers as input.

Each node n in the parse tree for a particular sentence is associated with a word w, a word vector x_w, and a hidden vector $h_n \in \mathbb{R}^d$ of the same dimension as the word vectors. For internal nodes, this vector is a phrase-level representation, while at leaf nodes it is the word vector x_w mapped into the hidden space. Unlike in constituency trees where all words reside at the leaf level, internal nodes of dependency trees are associated with words. Thus, the DT-RNN has to combine the current node's word vector with its children's hidden vectors to form h_n. This process continues recursively up to the root, which represents the entire sentence.

We associate a separate $d \times d$ matrix W_r with each dependency relation r in our dataset and learn these matrices during training.[1] Syntactically untying these matrices improves com-

[1] We had 46 unique dependency relations in our quiz bowl dataset.

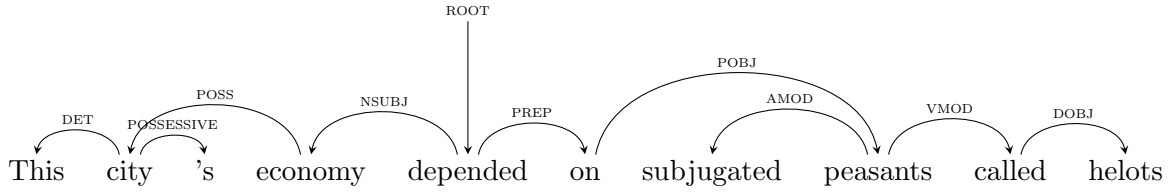

Figure 2: Dependency parse of a sentence from a question about Sparta.

positionality over the standard RNN model by taking into account relation identity along with tree structure. We include an additional $d \times d$ matrix, W_v, to incorporate the word vector x_w at a node into the node vector h_n.

Given a parse tree (Figure 2), we first compute leaf representations. For example, the hidden representation h_{helots} is

$$h_{\text{helots}} = f(W_v \cdot x_{\text{helots}} + b), \quad (1)$$

where f is a non-linear activation function such as tanh and b is a bias term. Once all leaves are finished, we move to interior nodes with already processed children. Continuing from "helots" to its parent, "called", we compute

$$h_{\text{called}} = f(W_{\text{DOBJ}} \cdot h_{\text{helots}} + W_v \cdot x_{\text{called}} + b). \quad (2)$$

We repeat this process up to the root, which is

$$h_{\text{depended}} = f(W_{\text{NSUBJ}} \cdot h_{\text{economy}} + W_{\text{PREP}} \cdot h_{\text{on}} + W_v \cdot x_{\text{depended}} + b). \quad (3)$$

The composition equation for any node n with children $K(n)$ and word vector x_w is $h_n =$

$$f(W_v \cdot x_w + b + \sum_{k \in K(n)} W_{R(n,k)} \cdot h_k), \quad (4)$$

where $R(n, k)$ is the dependency relation between node n and child node k.

3.2 Training

Our goal is to map questions to their corresponding answer entities. Because there are a limited number of possible answers, we can view this as a multi-class classification task. While a softmax layer over every node in the tree could predict answers (Socher et al., 2011; Iyyer et al., 2014), this method overlooks that most answers are themselves words (features) in other questions (e.g., a question on *World War II* might mention the *Battle of the Bulge* and vice versa). Thus, word vectors associated with such answers can be trained in the same vector space as question text,[2] enabling us to model relationships between answers instead of assuming incorrectly that all answers are independent.

To take advantage of this observation, we depart from Socher et al. (2014) by training both the answers and questions jointly in a single model, rather than training each separately and holding embeddings fixed during DT-RNN training. This method cannot be applied to the multimodal text-to-image mapping problem because text captions by definition are made up of words and thus cannot include images; in our case, however, question text can and frequently does include answer text.

Intuitively, we want to encourage the vectors of question sentences to be near their correct answers and far away from incorrect answers. We accomplish this goal by using a contrastive max-margin objective function described below. While we are not interested in obtaining a ranked list of answers,[3] we observe better performance by adding the weighted approximate-rank pairwise (WARP) loss proposed in Weston et al. (2011) to our objective function.

Given a sentence paired with its correct answer c, we randomly select j incorrect answers from the set of all incorrect answers and denote this subset as Z. Since c is part of the vocabulary, it has a vector $x_c \in W_e$. An incorrect answer $z \in Z$ is also associated with a vector $x_z \in W_e$. We define S to be the set of all nodes in the sentence's dependency tree, where an individual node $s \in S$ is associated with the

[2] Of course, questions never contain their own answer as part of the text.

[3] In quiz bowl, all wrong guesses are equally detrimental to a team's score, no matter how "close" a guess is to the correct answer.

hidden vector h_s. The error for the sentence is

$$C(S,\theta) = \sum_{s\in S}\sum_{z\in Z} L(rank(c,s,Z))\max(0, 1 - x_c \cdot h_s + x_z \cdot h_s), \quad (5)$$

where the function $rank(c,s,Z)$ provides the rank of correct answer c with respect to the incorrect answers Z. We transform this rank into a loss function[4] shown by Usunier et al. (2009) to optimize the top of the ranked list, $L(r) = \sum_{i=1}^{r} 1/i$.

Since $rank(c,s,Z)$ is expensive to compute, we approximate it by randomly sampling K incorrect answers until a violation is observed ($x_c \cdot h_s < 1 + x_z \cdot h_s$) and set $rank(c,s,Z) = (|Z|-1)/K$, as in previous work (Weston et al., 2011; Hermann et al., 2014). The model minimizes the sum of the error over all sentences T normalized by the number of nodes N in the training set,

$$J(\theta) = \frac{1}{N}\sum_{t\in T} C(t,\theta). \quad (6)$$

The parameters $\theta = (W_{r\in R}, W_v, W_e, b)$, where R represents all dependency relations in the data, are optimized using AdaGrad(Duchi et al., 2011).[5] In Section 4 we compare performance to an identical model (FIXED-QANTA) that excludes answer vectors from W_e and show that training them as part of θ produces significantly better results.

The gradient of the objective function,

$$\frac{\partial C}{\partial \theta} = \frac{1}{N}\sum_{t\in T}\frac{\partial J(t)}{\partial \theta}, \quad (7)$$

is computed using backpropagation through structure (Goller and Kuchler, 1996).

3.3 From Sentences to Questions

The model we have just described considers each sentence in a quiz bowl question independently. However, previously-heard sentences within the same question contain useful information that we do not want our model to ignore. While past work on RNN models have been restricted to the sentential and sub-sentential levels, we show that sentence-level representations can be easily combined to generate useful representations at the larger paragraph level.

The simplest and best[6] aggregation method is just to average the representations of each sentence seen so far in a particular question. As we show in Section 4, this method is very powerful and performs better than most of our baselines. We call this averaged DT-RNN model QANTA: a question answering neural network with trans-sentential averaging.

4 Experiments

We compare the performance of QANTA against multiple strong baselines on two datasets. QANTA outperforms all baselines trained only on question text and improves an information retrieval model trained on all of Wikipedia. QANTA requires that an input sentence describes an entity without mentioning that entity, a constraint that is not followed by Wikipedia sentences.[7] While IR methods can operate over Wikipedia text with no issues, we show that the representations learned by QANTA over just a dataset of question-answer pairs can significantly improve the performance of IR systems.

4.1 Datasets

We evaluate our algorithms on a corpus of over 100,000 question/answer pairs from two different sources. First, we expand the dataset used in Boyd-Graber et al. (2012) with publically-available questions from quiz bowl tournaments held after that work was published. This gives us 46,842 questions in fourteen different categories. To this dataset we add 65,212 questions from NAQT, an organization that runs quiz bowl tournaments and generously shared with us all of their questions from 1998–2013.

[4] Our experiments show that adding this loss term to the objective function not only increases performance but also speeds up convergence

[5] We set the initial learning rate $\eta = 0.05$ and reset the squared gradient sum to zero every five epochs.

[6] We experimented with weighting earlier sentences less than later ones in the average as well as learning an additional RNN on top of the sentence-level representations. In the former case, we observed no improvements over a uniform average, while in the latter case the model overfit even with strong regularization.

[7] We tried transforming Wikipedia sentences into quiz bowl sentences by replacing answer mentions with appropriate descriptors (e.g., "Joseph Heller" with "this author"), but the resulting sentences suffered from a variety of grammatical issues and did not help the final result.

Because some categories contain substantially fewer questions than others (e.g., astronomy has only 331 questions), we consider only literature and history questions, as these two categories account for more than 40% of the corpus. This leaves us with 21,041 history questions and 22,956 literature questions.

4.1.1 Data Preparation

To make this problem feasible, we only consider a limited set of the most popular quiz bowl answers. Before we filter out uncommon answers, we first need to map all raw answer strings to a canonical set to get around formatting and redundancy issues. Most quiz bowl answers are written to provide as much information about the entity as possible. For example, the following is the raw answer text of a question on the Chinese leader Sun Yat-sen: *Sun Yat-sen; or Sun Yixian; or Sun Wen; or Sun Deming; or Nakayama Sho; or Nagao Takano.* Quiz bowl writers vary in how many alternate acceptable answers they provide, which makes it tricky to strip superfluous information from the answers using rule-based approaches.

Instead, we use Whoosh,[8] an information retrieval library, to generate features in an active learning classifier that matches existing answer strings to Wikipedia titles. If we are unable to find a match with a high enough confidence score, we throw the question out of our dataset. After this standardization process and manual vetting of the resulting output, we can use the Wikipedia page titles as training labels for the DT-RNN and baseline models.[9]

65.6% of answers only occur once or twice in the corpus. We filter out all answers that do not occur at least six times, which leaves us with 451 history answers and 595 literature answers that occur on average twelve times in the corpus. These pruning steps result in 4,460 usable history questions and 5,685 literature questions. While ideally we would have used all answers, our model benefits from many training examples per answer to learn meaningful representations; this issue can possibly be addressed with techniques from zero shot learning (Palatucci et al., 2009; Pasupat and Liang, 2014), which we leave to future work.

We apply basic named entity recognition (NER) by replacing all occurrences of answers in the question text with single entities (e.g., *Ernest Hemingway* becomes *Ernest_Hemingway*). While we experimented with more advanced NER systems to detect non-answer entities, they could not handle multi-word named entities like the book *Love in the Time of Cholera* (title case) or battle names (e.g., *Battle of Midway*). A simple search/replace on all answers in our corpus works better for multi-word entities.

The preprocessed data are split into folds by tournament. We choose the past two national tournaments[10] as our test set as well as questions previously answered by players in Boyd-Graber et al. (2012) and assign all other questions to train and dev sets. History results are reported on a training set of 3,761 questions with 14,217 sentences and a test set of 699 questions with 2,768 sentences. Literature results are reported on a training set of 4,777 questions with 17,972 sentences and a test set of 908 questions with 3,577 sentences.

Finally, we initialize the word embedding matrix W_e with word2vec (Mikolov et al., 2013) trained on the preprocessed question text in our training set.[11] We use the hierarchical skip-gram model setting with a window size of five words.

4.2 Baselines

We pit QANTA against two types of baselines: bag of words models, which enable comparison to a standard NLP baseline, and information retrieval models, which allow us to compare against traditional question answering techniques.

BOW The BOW baseline is a logistic regression classifier trained on binary unigram indicators.[12] This simple discriminative model is an improvement over the generative quiz bowl answering model of Boyd-Graber et al. (2012).

[8] `https://pypi.python.org/pypi/Whoosh/`

[9] Code and non-NAQT data available at `http://cs.umd.edu/~miyyer/qblearn`.

[10] The tournaments were selected because NAQT does not reuse any questions or clues within these tournaments.

[11] Out-of-vocabulary words from the test set are initialized randomly.

[12] Raw word counts, frequencies, and TF-IDF weighted features did not increase performance, nor did adding bigrams to the feature set (possibly because multi-word named entities are already collapsed into single words).

BOW-DT The BOW-DT baseline is identical to BOW except we augment the feature set with dependency relation indicators. We include this baseline to isolate the effects of the dependency tree structure from our compositional model.

IR-QB The IR-QB baseline maps questions to answers using the state-of-the-art Whoosh IR engine. The knowledge base for IR-QB consists of "pages" associated with each answer, where each page is the union of training question text for that answer. Given a partial question, the text is first preprocessed using a query language similar to that of Apache Lucene. This processed query is then matched to pages uses BM-25 term weighting, and the top-ranked page is considered to be the model's guess. We also incorporate fuzzy queries to catch misspellings and plurals and use Whoosh's built-in query expansion functionality to add related keywords to our queries. **IR-WIKI** The IR-WIKI model is identical to the IR-QB model except that each "page" in its knowledge base also includes all text from the associated answer's Wikipedia article. Since all other baselines and DT-RNN models operate only on the question text, this is not a valid comparison, but we offer it to show that we can improve even this strong model using QANTA.

4.3 DT-RNN Configurations

For all DT-RNN models the vector dimension d and the number of wrong answers per node j is set to 100. All model parameters other than W_e are randomly initialized. The non-linearity f is the normalized tanh function,[13]

$$f(v) = \frac{\tanh(v)}{\|\tanh(v)\|}. \quad (8)$$

QANTA is our DT-RNN model with feature averaging across previously-seen sentences in a question. To obtain the final answer prediction given a partial question, we first generate a feature representation for each sentence within that partial question. This representation is computed by concatenating together the word embeddings and hidden representations averaged over all nodes in the tree as well as the root node's hidden vector. Finally, we send the average of all of the individual sentence features[14] as input to a logistic regression classifier for answer prediction.

FIXED-QANTA uses the same DT-RNN configuration as QANTA except the answer vectors are kept constant as in the text-to-image model.

4.4 Human Comparison

Previous work provides human answers (Boyd-Graber et al., 2012) for quiz bowl questions. We use human records for 1,201 history guesses and 1,715 literature guesses from twenty-two of the quiz bowl players who answered the most questions.[15]

The standard scoring system for quiz bowl is **10** points for a correct guess and **-5** points for an incorrect guess. We use this metric to compute a total score for each human. To obtain the corresponding score for our model, we force it to imitate each human's guessing policy. For example, Figure 3 shows a human answering in the middle of the second sentence. Since our model only considers sentence-level increments, we compare the model's prediction after the first sentence to the human prediction, which means our model is privy to less information than humans.

The resulting distributions are shown in Figure 4—our model does better than the average player on history questions, tying or defeating sixteen of the twenty-two players, but it does worse on literature questions, where it only ties or defeats eight players. The figure indicates that literature questions are harder than history questions for our model, which is corroborated by the experimental results discussed in the next section.

5 Discussion

In this section, we examine why QANTA improves over our baselines by giving examples of questions that are incorrectly classified by all baselines but correctly classified by QANTA. We also take a close look at some sentences that all models fail to answer correctly. Finally, we visualize the answer space learned by QANTA.

[13]The standard tanh function produced heavy saturation at higher levels of the trees, and corrective weighting as in Socher et al. (2014) hurt our model because named entities that occur as leaves are often more important than non-terminal phrases.

[14]Initial experiments with L_2 regularization hurt performance on a validation set.

[15]Participants were skilled quiz bowl players and are not representative of the general population.

	History			Literature		
Model	Pos 1	Pos 2	Full	Pos 1	Pos 2	Full
BOW	27.5	51.3	53.1	19.3	43.4	46.7
BOW-DT	35.4	57.7	60.2	24.4	51.8	55.7
IR-QB	37.5	65.9	71.4	27.4	54.0	61.9
FIXED-QANTA	38.3	64.4	66.2	28.9	57.7	62.3
QANTA	**47.1**	**72.1**	**73.7**	**36.4**	**68.2**	**69.1**
IR-WIKI	53.7	76.6	77.5	41.8	74.0	73.3
QANTA+IR-WIKI	**59.8**	**81.8**	**82.3**	**44.7**	**78.7**	**76.6**

Table 1: Accuracy for history and literature at the first two sentence positions of each question and the full question. The top half of the table compares models trained on questions only, while the IR models in the bottom half have access to Wikipedia. QANTA outperforms all baselines that are restricted to just the question data, and it substantially improves an IR model with access to Wikipedia despite being trained on much less data.

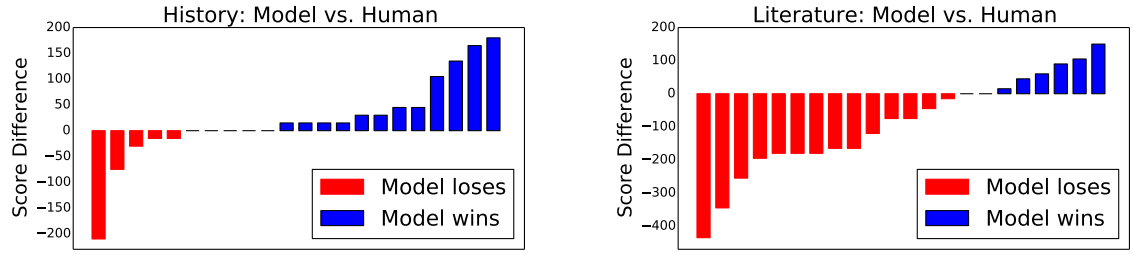

Figure 4: Comparisons of QANTA+IR-WIKI to human quiz bowl players. Each bar represents an individual human, and the bar height corresponds to the difference between the model score and the human score. Bars are ordered by human skill. Red bars indicate that the human is winning, while blue bars indicate that the model is winning. QANTA+IR-WIKI outperforms most humans on history questions but fails to defeat the "average" human on literature questions.

A minor character in this play can be summoned by a bell that does not always work; that character also doesn't have eyelids. Near the end, a woman who drowned her illegitimate child attempts to stab another woman in the Second Empire-style ◇ room in which the entire play takes place. For 10 points, Estelle and Ines are characters in which existentialist play in which Garcin claims "Hell is other people", written by Jean-Paul Sartre?

Figure 3: A question on the play "No Exit" with human buzz position marked as ◇. Since the buzz occurs in the middle of the second sentence, our model is only allowed to see the first sentence.

5.1 Experimental Results

Table 1 shows that when bag of words and information retrieval methods are restricted to question data, they perform significantly worse than QANTA on early sentence positions. The performance of BOW-DT indicates that while the dependency tree structure helps by itself, the compositional distributed representations learned by QANTA are more useful. The significant improvement when we train answers as part of our vocabulary (see Section 3.2) indicates that our model uses answer occurrences within question text to learn a more informative vector space.

The disparity between IR-QB and IR-WIKI indicates that the information retrieval models need lots of external data to work well at all sentence positions. IR-WIKI performs better than other models because Wikipedia contains many more sentences that partially match specific words or phrases found in early clues than the question training set. In particular, it is impossible for all other models to answer clues in the test set that have no semantically similar

or equivalent analogues in the training question data. With that said, IR methods can also operate over data that does not follow the special constraints of quiz bowl questions (e.g., every sentence uniquely identifies the answer, answers don't appear in their corresponding questions), which QANTA cannot handle. By combining QANTA and IR-WIKI , we are able to leverage access to huge knowledge bases along with deep compositional representations, giving us the best of both worlds.

5.2 Where the Attribute Space Helps Answer Questions

We look closely at the first sentence from a literature question about the author Thomas Mann: "He left unfinished a novel whose title character forges his father's signature to get out of school and avoids the draft by feigning desire to join".

All baselines, including IR-WIKI , are unable to predict the correct answer given only this sentence. However, QANTA makes the correct prediction. The sentence contains no named entities, which makes it almost impossible for bag of words or string matching algorithms to predict correctly. Figure 6 shows that the plot description associated with the "novel" node is strongly indicative of the answer. The five highest-scored answers are all male authors,[16] which shows that our model is able to learn the answer type without any hand-crafted rules.

Our next example, the first sentence in Table 2, is from the first position of a question on John Quincy Adams, which is correctly answered by only QANTA. The bag of words model guesses Henry Clay, who was also a Secretary of State in the nineteenth century and helped John Quincy Adams get elected to the presidency in a "corrupt bargain". However, the model can reason that while Henry Clay was active at the same time and involved in the same political problems of the era, he did not represent the Amistad slaves, nor did he negotiate the Treaty of Ghent.

5.3 Where all Models Struggle

Quiz bowl questions are intentionally written to make players work to get the answer, especially at early sentence positions. Our model fails to answer correctly more than half the time after hearing only the first sentence. We examine some examples to see if there are any patterns to what makes a question "hard" for machine learning models.

Consider this question about the Italian explorer John Cabot: "As a young man, this native of Genoa disguised himself as a Muslim to make a pilgrimage to Mecca".

While it is obvious to human readers that the man described in this sentence is not actually a Muslim, QANTA has to accurately model the verb *disguised* to make that inference. We show the score plot of this sentence in Figure 7. The model, after presumably seeing many instances of *muslim* and *mecca* associated with Mughal emperors, is unable to prevent this information from propagating up to the root node. On the bright side, our model is able to learn that the question is expecting a human answer rather than non-human entities like the Umayyad Caliphate.

More examples of impressive answers by QANTA as well as incorrect guesses by all systems are shown in Table 2.

5.4 Examining the Attribute Space

Figure 5 shows a t-SNE visualization (Van der Maaten and Hinton, 2008) of the 451 answers in our history dataset. The vector space is divided into six general clusters, and we focus in particular on the US presidents. Zooming in on this section reveals temporal clustering: presidents who were in office during the same timeframe occur closer together. This observation shows that QANTA is capable of learning attributes of entities during training.

6 Related Work

There are two threads of related work relevant to this paper. First, we discuss previous applications of compositional vector models to related NLP tasks. Then, we examine existing work on factoid question-answering and review the similarities and differences between these tasks and the game of quiz bowl.

6.1 Recursive Neural Networks for NLP

The principle of *semantic composition* states that the meaning of a phrase can be derived

[16] three of whom who also have well-known unfinished novels

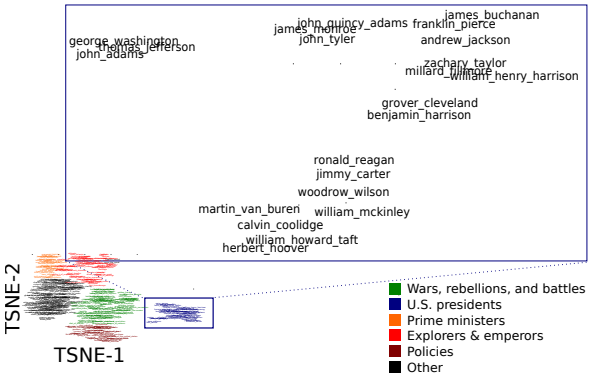

Figure 5: t-SNE 2-D projections of 451 answer vectors divided into six major clusters. The blue cluster is predominantly populated by U.S. presidents. The zoomed plot reveals temporal clustering among the presidents based on the years they spent in office.

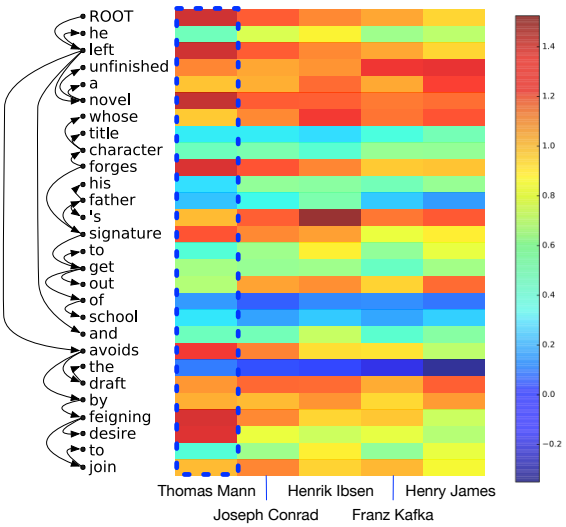

Figure 6: A question on the German novelist Thomas Mann that contains no named entities, along with the five top answers as scored by QANTA. Each cell in the heatmap corresponds to the score (inner product) between a node in the parse tree and the given answer, and the dependency parse of the sentence is shown on the left. All of our baselines, including IR-WIKI , are wrong, while QANTA uses the plot description to make a correct guess.

from the meaning of the words that it contains as well as the syntax that glues those words together. Many computational models of compositionality focus on learning vector spaces (Zanzotto et al., 2010; Erk, 2012; Grefenstette et al., 2013; Yessenalina and Cardie, 2011). Recent approaches towards modeling compositional vector spaces with neural networks have been successful, although simpler functions have been proposed for short phrases (Mitchell and Lapata, 2008).

Recursive neural networks have achieved state-of-the-art performance in sentiment analysis and parsing (Socher et al., 2013c; Hermann and Blunsom, 2013; Socher et al., 2013a). RNNs have not been previously used for learning attribute spaces as we do here, although recursive tensor networks were unsuccessfully applied to a knowledge base completion task (Socher et al., 2013b). More relevant to this work are the dialogue analysis model proposed by Kalchbrenner & Blunsom (2013) and the paragraph vector model described in Le and Mikolov (2014), both of which are able to generate distributed representations of paragraphs. Here we present a simpler approach where a single model is able to learn complex sentence representations and average them across paragraphs.

6.2 Factoid Question-Answering

Factoid question answering is often functionally equivalent to information retrieval. Given a knowledge base and a query, the goal is to return the answer. Many approaches to this problem rely on hand-crafted pattern matching and answer-type classification to narrow down the search space (Shen, 2007; Bilotti et al., 2010; Wang, 2006). More recent factoid QA systems incorporate the web and social media into their retrieval systems (Bian et al., 2008). In contrast to these approaches, we place the burden of learning answer types and patterns on the model.

7 Future Work

While we have shown that DT-RNNs are effective models for quiz bowl question answering, other factoid QA tasks are more challenging. Questions like *what does the* AARP *stand for?* from TREC QA data require additional infrastructure. A more apt comparison would be to IBM's proprietary Watson system (Lally et al., 2012) for Jeopardy, which is limited to single sentences, or to models trained on Yago (Hoffart et al., 2013).

We would also like to fairly compare QANTA

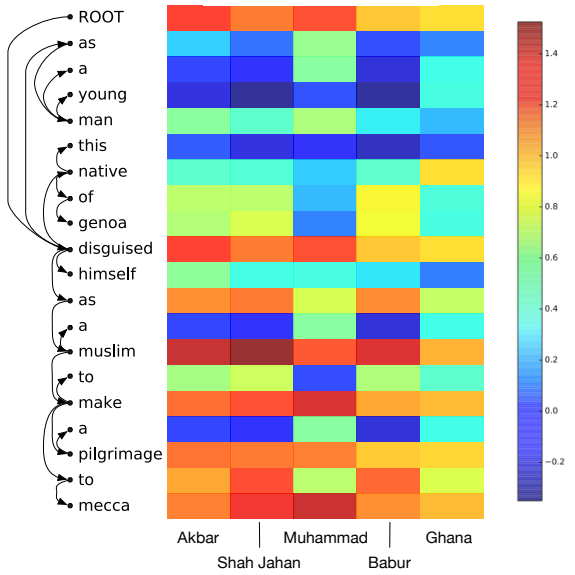

Figure 7: An extremely misleading question about John Cabot, at least to computer models. The words *muslim* and *mecca* lead to three Mughal emperors in the top five guesses from QANTA; other models are similarly led awry.

Q	he also successfully represented the amistad slaves and negotiated the treaty of ghent and the annexation of florida from spain during his stint as secretary of state under james monroe
A	**john quincy adams**, henry clay, andrew jackson
Q	this work refers to people who fell on their knees in hopeless cathedrals and who jumped off the brooklyn bridge
A	**howl**, the tempest, paradise lost
Q	despite the fact that twenty six martyrs were crucified here in the late sixteenth century it remained the center of christianity in its country
A	**nagasaki**, guadalcanal, ethiopia
Q	this novel parodies freudianism in a chapter about the protagonist 's dream of holding a live fish in his hands
A	billy budd, the ambassadors, all my sons
Q	a contemporary of elizabeth i he came to power two years before her and died two years later
A	grover cleveland, benjamin harrison, henry cabot lodge

Table 2: Five example sentences occuring at the first sentence position along with their top three answers as scored by QANTA; correct answers are marked with blue and wrong answers are marked with red. QANTA gets the first three correct, unlike all other baselines. The last two questions are too difficult for all of our models, requiring external knowledge (e.g., *Freudianism*) and temporal reasoning.

with IR-WIKI . A promising avenue for future work would be to incorporate Wikipedia data into QANTA by transforming sentences to look like quiz bowl questions (Wang et al., 2007) and to select relevant sentences, as not every sentence in a Wikipedia article directly describes its subject. Syntax-specific annotation (Sayeed et al., 2012) may help in this regard.

Finally, we could adapt the attribute space learned by the DT-RNN to use information from knowledge bases and to aid in knowledge base completion. Having learned many facts about entities that occur in question text, a DT-RNN could add new facts to a knowledge base or check existing relationships.

8 Conclusion

We present QANTA, a dependency-tree recursive neural network for factoid question answering that outperforms bag of words and information retrieval baselines. Our model improves upon a contrastive max-margin objective function from previous work to dynamically update answer vectors during training with a single model. Finally, we show that sentence-level representations can be easily and effectively combined to generate paragraph-level representations with more predictive power than those of the individual sentences.

Acknowledgments

We thank the anonymous reviewers, Stephanie Hwa, Bert Huang, and He He for their insightful comments. We thank Sharad Vikram, R. Hentzel, and the members of NAQT for providing our data. This work was supported by NSF Grant IIS-1320538. Boyd-Graber is also supported by NSF Grant CCF-1018625. Any opinions, findings, conclusions, or recommendations expressed here are those of the authors and do not necessarily reflect the view of the sponsor.

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *JMLR*.

Jiang Bian, Yandong Liu, Eugene Agichtein, and Hongyuan Zha. 2008. Finding the right facts in the crowd: factoid question answering over social media. In *WWW*.

Matthew W. Bilotti, Jonathan Elsas, Jaime Carbonell, and Eric Nyberg. 2010. Rank learning for factoid question answering with linguistic and semantic constraints. In *CIKM*.

Jordan Boyd-Graber, Brianna Satinoff, He He, and Hal Daume III. 2012. Besting the quiz master: Crowdsourcing incremental classification games. In *EMNLP*.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *LREC*.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 999999:2121–2159.

Katrin Erk. 2012. Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*.

Christoph Goller and Andreas Kuchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *Neural Networks, 1996., IEEE International Conference on*, volume 1.

Edward Grefenstette, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. 2013. Multi-step regression learning for compositional distributional semantics. *CoRR*.

Karl Moritz Hermann and Phil Blunsom. 2013. The Role of Syntax in Vector Space Models of Compositional Semantics. In *ACL*.

Karl Moritz Hermann, Edward Grefenstette, and Phil Blunsom. 2013. "not not bad" is not "bad": A distributional account of negation. *Proceedings of the ACL Workshop on Continuous Vector Space Models and their Compositionality*.

Karl Moritz Hermann, Dipanjan Das, Jason Weston, and Kuzman Ganchev. 2014. Semantic frame identification with distributed word representations. In *ACL*.

Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 194:28–61.

Mohit Iyyer, Peter Enns, Jordan Boyd-Graber, and Philip Resnik. 2014. Political ideology detection using recursive neural networks.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent convolutional neural networks for discourse compositionality. *Proceedings of the 2013 Workshop on Continuous Vector Space Models and their Compositionality*.

Adam Lally, John M Prager, Michael C McCord, BK Boguraev, Siddharth Patwardhan, James Fan, Paul Fodor, and Jennifer Chu-Carroll. 2012. Question analysis: How watson reads a clue. *IBM Journal of Research and Development*.

Quoc V Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *ACL*.

Mark Palatucci, Dean Pomerleau, Geoffrey E. Hinton, and Tom M. Mitchell. 2009. Zero-shot learning with semantic output codes. In *NIPS*.

P. Pasupat and P. Liang. 2014. Zero-shot entity extraction from web pages. In *ACL*.

Asad B Sayeed, Jordan Boyd-Graber, Bryan Rusk, and Amy Weinberg. 2012. Grammatical structures for word-level sentiment detection. In *NAACL*.

Dan Shen. 2007. Using semantic role to improve question answering. In *EMNLP*.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In *EMNLP*.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013a. Parsing With Compositional Vector Grammars. In *ACL*.

Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013b. Reasoning With Neural Tensor Networks For Knowledge Base Completion. In *NIPS*.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013c. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*.

Richard Socher, Quoc V Le, Christopher D Manning, and Andrew Y Ng. 2014. Grounded compositional semantics for finding and describing images with sentences. *TACL*.

Nicolas Usunier, David Buffoni, and Patrick Gallinari. 2009. Ranking with ordered weighted pairwise classification. In *ICML*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *JMLR*.

Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? a quasi-synchronous grammar for QA. In *EMNLP*.

Mengqiu Wang. 2006. A survey of answer extraction techniques in factoid question answering. *Computational Linguistics*, 1(1).

Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*.

Ainur Yessenalina and Claire Cardie. 2011. Compositional matrix-space models for sentiment analysis. In *EMNLP*.

Fabio Massimo Zanzotto, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar. 2010. Estimating linear models for compositional distributional semantics. In *COLT*.