Economic Worth-Aware Word Embeddings

Yusan Lin
Visa Research
California, United States
yusalin@visa.com

Peifeng Yin
Pinterest
California, United States
pyin@pinterest.com

Wang-Chien Lee
The Pennsylvania State University
Pennsylvania, United States
wlee@cse.psu.edu

***Abstract*—Knowing the perceived economic value of words is often desirable for applications such as product naming and pricing. However, there is a lack of understanding on the underlying *economic worths* of words, even though we have seen some breakthrough on learning the semantics of words. In this work, we bridge this gap by proposing a joint-task neural network model, Word Worth Model (WWM), to learn word embedding that captures the underlying economic worths. Through the design of WWM, we incorporate contextual factors, e.g., product's brand name and restaurant's city, that may affect the aggregated monetary value of a textual item. Via a comprehensive evaluation, we show that, compared with other baselines, WWM accurately predicts missing words when given target words. We also show that the learned embeddings of both words and contextual factors reflect well the underlying economic worths through various visualization analyses.**

***Index Terms*—economic worth, word worth, word embedding learning**

I. Introduction

The learning of *word embeddings*, i.e., vector representations of words, has attracted much attention due to its applaudable intent to capture both semantic and syntactic meanings of words in terms of real-number vectors and its ubiquitous application base [1]–[4]. The success of word2vec [3] shed a light in this important research direction, leading to many variants [5]–[7] and their applications in text mining and natural language processing [8]–[11]. While these prior works stick to the original intent of "capturing semantic and synthetic meaning", we argue that the goal of learning word embeddings can be further extended to learn the *economic worth* [12] of words, which we envisage to have a great number of potential applications in a wide range of domains.

We define the economic worth of a word, namely *word worth*, as the associated monetary value. In some cases, a word stands for a real object and the word worth reflects that object's price in the market. One example comes from restaurant menus. As shown in Figure 1, *steak* and *salmon* are more expensive than *egg* and *cheese* on average[1]. In other cases, a word represents an abstract concept and its worth is closer to a more complex *perceived value* [13]. One example is the *brand value*, where "Valentino" always impresses people with luxury while "Old Navy" is often linked to affordable items, as shown in Figure 2(a).

Learning the word worths have several benefits. For example, one may predict the prices of products, menu dishes, and sales packages with given descriptions or titles based on the learned word worths. Similarly, it may help recommend appropriate words to describe products with a targeted price, e.g., by formulating a missing word prediction problem, given the listed prices and other words used to describe the products or dishes. This idea can be further extended to help recommend appropriate words to describe products with a targeted price. Secondly, learning the word worth enables classification tasks that are word-worth-driven. For example, with only the restaurant's menu provided, one may be able to categorize the price range the restaurant lies in, which might be a difficult task with only the understanding of words' semantics.

Learning the word worths is a challenging task, as the perceived value itself is highly variable, dependent on various factors. Firstly, the underlying worths of words may benefit from a *multi-dimensional* representation, which is where the approach proposed by Chahuneau falls short [14]. Figure 3 shows the price distribution of four different words in a restaurant menu dataset. As shown, for each word, the prices it is associated with spread across a wide range. Secondly, the economic worth is *context-dependent*. For instance, the word

[1]Each word's price is the average prices of all the dishes the word has been used to describe.

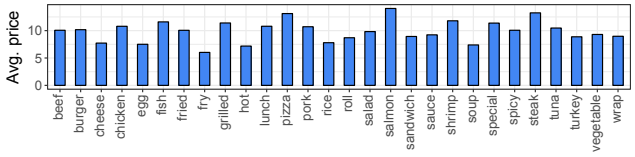
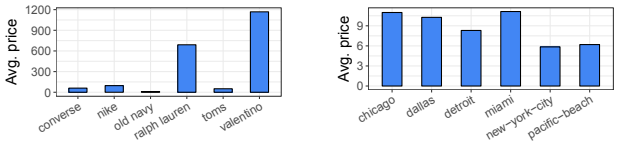
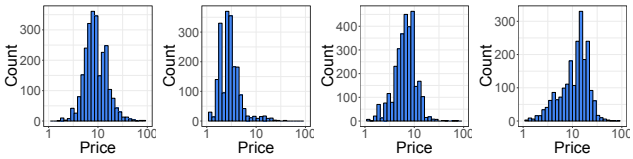


Fig. 1. Average prices of words in menus



(a) Shoe brands (b) Cities

Fig. 2. Average prices of different contextual factors



(a) Tuna (b) Tea (c) Soup (d) Pizza

Fig. 3. Price distribution of words in restaurant menus

978-1-7281-8206-3/20/$31.00 ©2020 IEEE
DOI 10.1109/DSAA49011.2020.00048

tuna is not expensive when it is associated with sandwiches, yet it is highly priced in Japanese restaurants when served as sashimi. Another context is location. For example, Figure 2(b) shows the average price of menu items for restaurants in different US cities, which varies a lot due to the demand-supply equilibrium. In summary, word worth is a complicated concept and is hard to capture fully.

To encounter the characteristics of word worth and challenges mentioned above, we propose a joint-task neural network model, *word worth model (WWM)*, that learns the representations of words which, in addition to the semantics, capture the word worths. The proposed model incorporates contextual factors that affect the word worths, such as locations and brands.

To achieve the above, WWM aims to learn two tasks as a joint-task neural network model: (1) *missing word prediction*, and (2) *price prediction* by optimizing two different objective functions, *semantic objective* and *economic-worth objective*, respectively. Furthermore, we design the *contextual bias integration*, which addresses the context-dependent characteristic of word worths. To enable the ability of joint-task learning, we propose two input sampling methods that activate the two objective functions: *subsequence sampling* and *whole sequence sampling*. When WWM is provided with subsequence sampled input, it activates the missing word prediction task, which aims to optimize the semantic objective; when WWM is provided with the whole sequence sampled input, it activates the price prediction task, hence optimizing the economic-worth objective. We train WWM on four different datasets that associate textual descriptions of objects and prices.

Quantitatively, through such a design, compared with representations learned by naive approaches, the word embeddings learned by WWM are shown to better predict the product prices, given the words in product descriptions, and better predict the other words to describe the products when given part of the descriptions.

Qualitatively, the learned word embedding and contextual biases are also shown to capture well the underlying economic worth when demonstrated through visualization. More specifically, we visualize the learned word embeddings on a t-SNE-transformed, two-dimensional space, and find that words are distributed closely together not only when they have similar semantic meanings, but also when they associate with similar range of monetary values. Also, when visualizing the different contextual biases' effects on the same word, one can see that we are able to capture the luxury level of brands, and cost of living of cities.

The contributions of this work are summarized below.

1) To the best knowledge of the authors, this is the first work attempting to learn the word embedding capturing economic worths, which have many potential applications in a wide range of domains.
2) We propose a joint-task neural network model, word worth model (WWM) that learns word embeddings capturing the economic worths and semantics by jointly optimizing two tasks: missing word prediction and price prediction. We also propose two input sampling methods that enable the two tasks. Quantitatively, WWM is shown to better predict product prices given descriptions, and better predict missing words given part of the descriptions. Qualitatively, the word embeddings learned by WWM are shown to better capture words that are similar in terms of economic worths, and the contextual biases can capture the effects of different factors on the final prices.
3) We implemented and open-sourced the proposed word worth model (WWM).[2] We believe through WWM, many potential applications can be enabled, including product naming, menu design, and pricing.

The rest of the paper is organized as follows. We first review the literature in section II. We conduct data analyses to show the word worths in section III. Our proposed word worth model (WWM) and input sampling methods are presented in section IV. In the section V , we evaluate the proposed models with three real-world datasets, and demonstrate the ability of the representations learned by our models in price prediction and word prediction. We also show the quality of the learned word embeddings through various visualization analyses.Finally, we conclude this work and discuss future works in section VI.

II. Related Work

We review two lines of research relevant to this work: 1) the economic worths of words, and 2) representation learning.

A. Economic Worths of Words

The notion of "economic worths of words", or *word worths*, is first mentioned by Kuehling [12], where the change in the value of words is exploited to understand the "tides of innovation". However, there has been little follow-up research after his initiation. The word worths can be expressed in different ways besides monetary values. For example, in marketing, it is measured as *customers' perceived value*. Eggert et al. first raise the idea of measuring the customer perceived value, and claim that it is a better measurement than customer satisfaction for research on business marketing [15]. Yang et al. also argue that companies interested in customer loyalty should focus on raising their perceived values [16].

In this work, we focus on the word worths as monetary value. Chahuneau et al. leverage natural language processing and linear regression to predict menu items' prices based on their textual descriptions [14]. In this approach, the word worths are represented by corresponding coefficients of the learned regression model. However, as we show later in Figure 3, the word worth should be multi-dimensional. Modeling it as a scalar coefficient leads to a limited understanding.

B. Representation Learning

Representation learning of different items, e.g., words, images, and graphs, has long been studied [17]. In [3], Mikolov et. al. propose neural network models, known as *word2vec*, to learn word representations. Via unsupervised learning, it exploits the co-occurrence patterns of words within a predefined context window. Le et. al. [18] later extends word2vec to

[2]https://github.com/yusanlin/word-worth-embedding/

capture the representations of sentences as well as documents (i.e., *doc2vec*).

Inspired by the elegant yet powerful design of word2vec, numerous models and sampling methods are developed to capture the nodes' and edges' representations in networks [19]–[21]. Some works further capture the representations of meta paths [22], [23]

The design of typical representation learning methods is constrained by *categorical* learning targets. For targets with continuous values, the problem is usually transformed into approximated discrete labels, such as [24]. In this work, we aim to learn a word representation that reflects its word worth of a continuous value.

III. Data Analysis

In this section we conduct a data analysis on menus to demonstrate the characteristics of word worths in terms of the following three characteristics: i) multi-dimensionality and ii) context dependency.[3]

A. Multi-Dimensionality

Chahuneau et al. learn word worths by fitting linear regression models, where the word worths can be obtained from the learned coefficients [14]. Here we follow the above-mentioned approach to investigate our data. Specifically, we fit a linear regression model using the 100 most frequent words in restaurant menus as the independent variables, and the prices of menu items as dependent variable. The result is shown in Table I.[4] As shown in the first row, the average price of menu items in the dataset is $10.54 USD (the intercept). Depending on words used in the menu, the estimated price is updated with the corresponding coefficient. For example, the price of a dish *fried chicken* would most likely be $\$10.54 + 0.22 + 0.38 = 11.14$ USD.

TABLE I
LINEAR REGRESSION RESULTS OF RESTAURANT MENU DATA

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	10.53678	0.06423	164.041	<2e-16	***
chicken	0.38063	0.0886	4.296	1.74E-05	***
salad	-0.71092	0.1086	-6.546	5.93E-11	***
sandwich	-1.50568	0.12507	-12.039	<2e-16	***
cheese	-2.28567	0.12966	-17.628	<2e-16	***
beef	-0.1215	0.14208	-0.855	0.392494	
shrimp	1.31539	0.14271	9.217	<2e-16	***
roll	-2.08282	0.14589	-14.277	<2e-16	***
lunch	0.04305	0.15508	0.278	0.781315	
fried	0.22017	0.16455	1.338	0.180901	
rice	-2.91843	0.17412	-16.761	<2e-16	***
egg	-2.32797	0.16066	-14.49	<2e-16	***
grilled	1.08832	0.16378	6.645	3.05E-11	***
soup	-3.07668	0.19335	-15.912	<2e-16	***
pork	0.67305	0.17325	3.885	0.000102	***
special	0.97969	0.17758	5.517	3.46E-08	***

Although most of the words presented in the result significantly contribute to the price (see the *** in the last column), this approach of estimating the word worths suffers from one major drawback: it only captures words' average associated prices and loses meaningful context information. As Figure 3 in the section I suggests, the distribution of prices with respect to a word in the menus varies greatly. Also, for different words, the distributions of their prices are also very different.

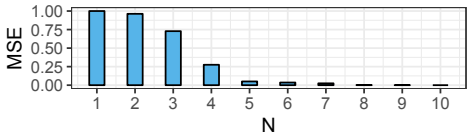
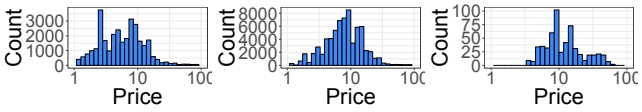


Fig. 4. MSE of price prediction with different dimensions

Figure 4 shows a comparison of price prediction when word worth is modeled as multi-dimensional vectors vs. a scalar (conducted using model later shown in Figure 8(b)). In the figure, as N increases, the MSE in capturing prices of given words decreases. Hence confirms the strength of using multi-dimensional representation of words to capture economic worths.

B. Context Dependency

Besides having a multi-dimensional representation of word worth, some *contextual factors* can also affect the prices. One example is the location: restaurants and their menus may have significantly different prices due to the diversified living cost of their located cities. As shown in Figure 5, the item prices in restaurant menus differ across cities.



(a) Indianopolis (b) New York (c) Coronado

Fig. 5. Price distribution of cities in restaurant menus

To sum up, the word worth has two unique characteristics, i.e., multi-dimensionality and context dependency. Furthermore, when learning the representation of word worth, we do not want to lose the semantics of these words. In the next section, we develop the word worth model that learns the word embedding carrying the underlying semantic meaning and economic worths, considering the multi-dimensionality and context dependency.

IV. Word Worth Model

In this section, we describe our *Word Worth Model (WWM)* to learn the underlying economic worth and semantics of words. We first give an overview of the model structure and then discuss model training. For ease of explanation, we summarize the symbols and corresponding definitions used in this paper in Table II.

[3]The detailed dataset statistics will be discussed in section V

[4]Due to space limitation, we only show the results of the most frequent 15 words

TABLE II
SYMBOL DEFINITIONS

Symbol	Definition
s	raw input sequence (words)
T	sequence length
N	dimension of both word embedding and RNN hidden layer
V	vocabulary size
$\mathbf{W}^e$	$V \times N$ matrix of word embedding
$\mathbf{h}^s$	embedding vector for a textual sequence (output of RNN)
$\mathbf{W}^r_{hh}$	$N \times N$ matrix, RNN parameter
$\mathbf{W}^m$	$N \times V$ matrix for missing-word prediction
$\mathbf{p}^m$	V-dimensional probability vector of missing words
L	number of contextual factors
$\mathbf{C}$	L-dimensional binary vector for contextual factors
$\mathbf{b}^c$	N-dimensional vector for contextual bias
$\mathbf{W}^c$	$L \times N \times N$ contextual matrix
$y, \tilde{y}$	real and predicted price
$\mathbf{W}^y$	$N \times 1$ matrix of price predictor
$\mathbb{D}$	data set
$\tilde{s}$	sub-sequence
$\mathbb{S}$	set of all possible sub-sequences
$\mathbf{\Theta}$	all model parameters
$\mathcal{J}_s(\mathbf{\Theta})$	semantic-related objective function
$\mathcal{J}_w(\mathbf{\Theta})$	economic-worth objective function
α	balance control of whole and sub-sequence sampling

A. Model Overview

Figure 6 shows the structure of WWM. There are three major components in the model, including *missing word prediction*, *price prediction* and *contextual integration* (see dashed boxes in the figure). All of the them rely on a Recurrent Neural Network (RNN) to encode a raw textual sequence to a hidden numerical representation.

As shown in Figure 6, the computing flow of the model has two steps: 1) sequence encoding and 2) target prediction. For the first step, denoted as *Step 1* in the red shaded area in Figure 6, given a sequence $s = \langle w_1, \cdots, w_T \rangle$ of length T, the embedding matrix $\mathbf{W}^e$ is used to convert each raw token (represented as one-hot vector) to its corresponding embedding $\langle \mathbf{x}_1, \cdots, \mathbf{x}_T \rangle$. Then an RNN is used to generate the hidden states of each word in the sequence as below.

$$\begin{aligned} \mathbf{h}_0 &= \vec{\mathbf{0}} \\ \mathbf{h}_t &= \mathbf{x}_t + \mathbf{W}^r_{hh} \cdot \mathbf{h}_{t-1} \end{aligned} \tag{1}$$

Here we customize the RNN by i) setting the hidden dimension to be equivalent to the embedding dimension, and ii) removing the conventional input-to-hidden matrix. The purpose is to exploit directly the embedding into the follow-up prediction. In an extreme case where the sequence contains only one token, the sequence embedding $\mathbf{h}^s$ would be exactly the token embedding.

There are two branches in the second step, denoted by the two blue areas in Figure 6, of which one is to predict the missing tokens (labeled as *Step 2(a)*) and the other is to predict the price (labeled as *Step 2(b)*). As mentioned earlier in the data analysis, both the missing words and prices may depend on the associating contextual factors. We assume there is a limited number of contextual factors and each factor may or may not be active. Formally, let L denote the total number of contextual factors, a context can then be represented as

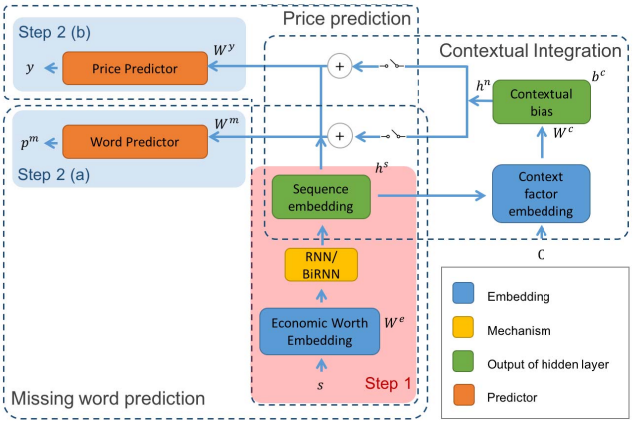


Fig. 6. Word worth model

a L-dimensional vector $\mathbf{C}$, where each element is a binary value indicating whether the corresponding factor is active or not. Given a sequence embedding $\mathbf{h}^s$ and its context $\mathbf{C}$, the contextual bias $\mathbf{b}^c$ can be obtained as follows.

$$\mathbf{b}^c = \mathbf{h}^s \cdot (\mathbf{C} \odot \mathbf{W}^c) \tag{2}$$

Here the $\mathbf{W}^c$ is a $L \times N \times N$ matrix encapsulating all contextual impacts on an item's economic worth. The contextual vector $\mathbf{C}$ determines which effect is activated. The operator $\odot$ is to select the weight of each active impact and sum them up, as defined below.

$$\mathbf{C} \odot \mathbf{W}^c = \sum_{l=1}^{L} \mathbf{C}[l] \cdot \mathbf{W}^c[l] \tag{3}$$

Particularly, if the context is not available, i.e., the vector $\mathbf{C}$ is all zero and the resulted bias is a zero-vector.

However, whether the contextual factors affect the missing words and prices may vary across different datasets. To control the different effects of contextual factors on missing words and prices, we design *switches*. There are two switches in WWM: *semantic switch*, q_s, and *economic-worth switch*, q_w. The two switches control how contextual factors are incorporated with sequence embedding h_s to predict missing words and prices differently.

The structure of missing-word prediction is similar to that of word2vec. Given a partial sequence vector, the model aims to predict missing words that usually co-occur with those existing ones. Formally, let $\mathbf{h}^s$ denote the sequence's vector and $\mathbf{W}^m$ represent the predictor's matrix parameters. The probability vector $\mathbf{p}^m$ over the vocabulary is shown as the equation below.

$$\mathbf{p}^m = \begin{cases} \texttt{softmax}\left((\mathbf{h}^s + b^c) \cdot \mathbf{W}^m\right), & \text{if } q_s = 1 \\ \texttt{softmax}\left(\mathbf{h}^s \cdot \mathbf{W}^m\right), & \text{otherwise.} \end{cases} \tag{4}$$

The prediction of the price aims to estimate the economic worth of the given sequence. The final price y is predicted based on the economic-worth switch q_w as below.

$$\tilde{y} = \begin{cases} \texttt{relu}\big((\mathbf{h}^s + \mathbf{b}^c) \cdot \mathbf{W}^y\big), & \text{if } q_w = 1 \\ \texttt{relu}\big(\mathbf{h}^s \cdot \mathbf{W}^y\big), & \text{otherwise.} \end{cases} \tag{5}$$

where $\texttt{relu}(x) = \max\{x, 0\}$.

B. Model Training

The goal of our model is to learn the word embeddings that capture both semantic meaning and economic worth. Therefore, the training of the model takes two objective functions into consideration. We assume that in a dataset $\mathbb{D}$, each instance consists of three elements: raw textual sequence s, the item price y_s and its context $\mathbf{C}_s$. For semantic-related objective function, only the raw text sequence is used, while for economic-worth, all three elements are involved.

Semantic Objective. Given a sequence s, let $\mathbb{S}$ denote the set of all possible sub-sequences. A sub-sequence $\tilde{s}$ is a partial word sequence of the original one. In prediction of missing words, our model gives a probability vector $\mathbf{p}^m$. The semantic objective function is to minimize the negative log-likelihood of corresponding real missing words. Formally, let $\mathbf{\Theta} = \{\mathbf{W}^e, \mathbf{W}^r_{hh}, \mathbf{W}^m, \mathbf{W}^c, \mathbf{W}^y\}$ denote all model parameters, the semantic objective function $\mathcal{J}_s(\mathbf{\Theta})$ is as follows.

$$\mathcal{J}_s(\mathbf{\Theta}) = -\frac{1}{|\mathbb{D}|}\sum_{s\in\mathbb{D}}\frac{1}{|\mathbb{S}|}\sum_{\tilde{s}\in\mathbb{S}}\sum_{w\in s/\tilde{s}}\log \mathbf{p}^m_{\tilde{s}}[w] \tag{6}$$

where $s/\tilde{s}$ represents a set of words that appear in the original sequence s but missing in the sub-sequence $\tilde{s}$. Also, $\mathbf{p}^m_{\tilde{s}}[w]$ denotes the probability of the corresponding missing word estimated by our model.

Economic-worth Objective. As shown in the model structure (Figure 6), besides the probability of missing words, another output is the estimated price for the whole sequence. The economic-worth objective function is the mean squared error (MSE) between the predicted price and the true one.

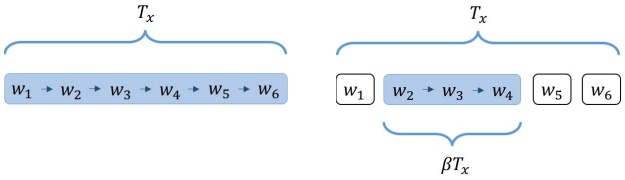
$$\mathcal{J}_w(\mathbf{\Theta}) = \frac{1}{|\mathbb{D}|}\sum_{\langle s, y_s, \mathbf{C}_s\rangle\in\mathbb{D}}(y_s - \tilde{y}_s)^2 \tag{7}$$

The final objective function is the sum of semantic and economic-worth. Here we do not introduce any parameter in merging the two objective functions even though the two may be in different scales. Based on Equations (6) and (7), the two objective functions are not active at the same time. Specifically, the semantic objective function $\mathcal{J}_s(\mathbf{\Theta})$ is only activated when the input is a sub-sequence while the economic-worth objective function $\mathcal{J}_w(\mathbf{\Theta})$ is activated only when the input is a whole sequence.

In reality, it is impossible to traverse all possible sub-sequences as it is quadratically growing with the data size. To address this, we propose a mechanism of input sampling to i) accelerate the training, and ii) take advantage of the fact that the two functions are not in play at the same time.

1) Input Sampling: We propose two types of sampling methods as shown in Figure 7. Here we leverage a toy example for illustration. Let an object of value that is worth y be described by $s = \langle w_1, w_2, w_3, w_4, w_5, w_6\rangle$. The two sampling methods prepare inputs for the model as follows.

Whole-sequence sampling. This sampling method takes in the whole description $s = \langle w_1, w_2, w_3, w_4, w_5, w_6\rangle$ as an input sequence and provides the ground truth price y. It thus activates the economic-worth objective $\mathcal{J}_w(\mathbf{\Theta})$ for the training to minimize. Formally, we denote the sequence s as $(w_i)_{1\le i\le T}$, where T is the length of sequence s.



(a) Whole sequence (b) Sub-sequence

Fig. 7. Input sampling methods.

Sub-sequence sampling. This sampling method also produces two inputs: sub-sequence $\tilde{s}$ and missing words $s/\tilde{s}$. In some case, the sub-sequence may include only a single word, i.e., the shaded word w_2 shown as the first example in Figure 7(b). In this case, the missing word is one of those in the solid-circled boxes, i.e., $\{w_1, w_3, w_4, w_5, w_6\}$. Alternatively, it may include a sequence of words with length shorter than s, as the second example in Figure 7(b) show, i.e., the sub-sequence words are $\langle w_2, w_3, w_4\rangle$, while the missing word is among $\{w_1, w_5, w_6\}$. Formally, the sub-sequence $\tilde{s}$ of s can be denoted as $(w_i)_{t_1\le i\le t_2}$, where $1 \le t_1 \le t_2 \le T$, and the length of the sub-sequence is βT. With the selected sub-sequence $\tilde{s}$, missing words are hence $s/\tilde{s} = \{w_i | 1 \le i < t_1 \vee t_2 < i \le T\}$.

The model learns words' semantics and economic worths by iteratively predicting the corresponding price and missing words of the given sequence, with inputs provided by the two sampling methods discussed above.

To train the proposed word worth model, there are several hyper parameters to choose: the N for the hidden dimension, the parameter controlling the ratio between optimizing semantic objective and economic-worth objective, and the parameter determining the lengths of the sub-sequences, which we discuss further in section V.

Let $0 \le \alpha \le 1$, where α is the probability that an epoch optimizes the economic-worth objective, and $1 - \alpha$ is the probability that an epoch optimizes the semantic objective. During training, in every epoch, an α-biased coin is tossed to decide objective to optimize. If it is the head, then the inputs sampled by whole sequence sampling are sent into WWM, which optimizes the worth objective $\mathcal{J}_w(\mathbf{\Theta})$. If it is the tail, then the inputs sampled by sub-sequence sampling is sent into WWM, which optimizes the semantic objective $\mathcal{J}_s(\mathbf{\Theta})$.

V. Evaluation

We evaluate the learned embedding both qualitatively and quantitatively. For qualitative analyses, we investigate that given a word, the distribution of the associated prices of its nearest neighbors in the embedding space. We then investigate how, in different datasets, the words locate in the high-dimensional space through visualizations of the t-SNE-transformed embeddings. We also analyze how the contextual biases affect the words' embedding in the space, and how that reflect on their associated prices. For quantitative analyses, we design two prediction tasks: price prediction and word prediction. We compare the performances of WWM with

four baselines, including word2vec, word-price model (WP), missing-word-price model (MWP), and missing-word-context-price model (MWCP). The first is the state-of-the-art model for learning word embedding, and the rest are feed forward neural network models we explore in our preliminary investigation of this research, which we show the architecture of these baseline models in Figure 8.

A. Datasets and Experiment Settings

We use four datasets, each of which represents a different scenario where words are associated with economic worths. Table III shows a summary of these datasets. All models are implemented using PyTorch, and we released the implementation of WWM and the datasets on Github.[5] For all models including baselines, we set the hidden dimension for both word embedding and contextual biases to 128. We then set the joint-task ratio $\alpha = 0.5$, and the sampling parameter $\beta = 0.5$. All of the weights in the networks are initialized using Xavier initialization. All models are learned using a batch size of 256 instances, and the learning stops when the loss functions converge, or until the maximum number of epochs is reached, i.e., 100.

TABLE III
DATASET SUMMARY

Dataset	Description	# of records	Object	Factor
Menu	Restaurant menus	797,798	Dish	City
Shoe	Shoe store listing	26,285	Shoe	Brand
Retail	Online store listing	541,909	Product	Country
Reward	Crowdfunding projects	636,673	Reward	Category

The performance metrics we use to evaluate the models' performances in our evaluation are as follows. For price prediction, since it is a continuous prediction target, we use mean squared error (MSE) to measure the performance. For missing word prediction, we use precision and recall to measure the performance.

B. Baselines

To evaluate how well WWM captures the words' underlying semantic and economic-worth compared with other models, we compare the word embedding learned by WWM with four baseline neural network models shown in Figure 8 (see next page).

Word2vec is the state-of-the-art word embedding learning model proposed in [3]. It takes in words used to describe an object, transforms the words into low-dimensional representations, and predicts the co-occurring words.

Word-price model (WP) takes in words used to describe an object, transforms the input into low-dimensional representations, and predicts the price of the object.

Missing-word-price model (MWP) extends the popular word2vec structure to include price information during embedding learning.It takes in a target word, encodes and transforms it into a low-dimensional representation, which then predicts the missing words and the price.

Missing-word-context-price (MWCP), besides the missing words and corresponding price, incorporates the contextual bias as a normalization factor, inspired by doc2vec [18].The model takes in the target words and contextual factor, transforms both of them into the low-dimensional representations, and predicts both the missing words and price of the object.

C. Learned Word Embedding

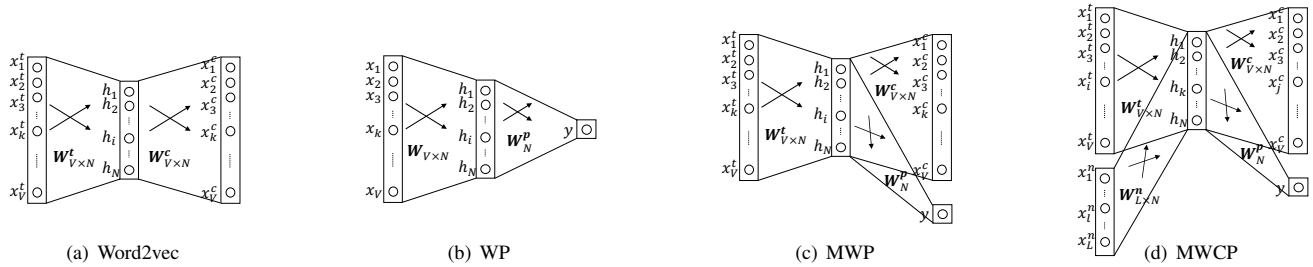
For word embeddings learned by each model on each dataset, we show the word embedding correlation with associated prices. More specifically, for each dataset, we select a word (*lobster* for menu, *suede* for shoes, *homemade* for retail, and *jacket* for reward) and use its top 10 similar words according to the learned word embeddings, and plot the price distributions of the words in the dataset.[6] The results are shown in Figures 9, 10, 11, and 12.

As shown, as the model considers more aspect of word worths, the less variation the prices words associated with get. In particular, the significant difference between WWM and word2vec on menu dataset truly demonstrates the capability of WWM in learning economic-worth-aware embeddings. However, one can also see that there are times the quality of embeddings produced by WWM is less ideal. For example, for the retail and reward datasets, the improvement of reducing price variances are smaller thant in menu and shoe datasets. For the retail dataset, we figure it is due to the fact that within the dataset, the ranges of product categories and prices are both too wide. As for the reward dataset, we believe that the nature of the reward description in crowdfunding projects are not as concise as product names in other datasets. Hence the uninformative words create noise for WWM to capture word worths.

Another aspect we look at is how the word embedding locates in two-dimensional space. We first transform the learned word embeddings from 128-dimensional to two-dimensional vectors by applying tSNE. We then visualize them as scatter plots, along with their associated average prices on the menu and shoe datasets, in Figures 13 and 14, respectively. We also compare the visualization of embeddings learned by word2vec and WWM. As one can see that in Figure 13, as one traverses towards the bottom-right in the visualization of WWM, the prices of words increase almost consistently. In particular, the words *champagne* and *caviar*, which both associate with expensive dishes, are located in the bottom-right corner, while the cheaper words such as *butter* and *gravy* are located on the opposite side of the plot. In contrary, when looking at the visualization generated using word2vec's embedding, one can see that words associated with different price range are meshed together in the plot. Similar phenomenon can be observed on the shoe dataset, as shown in Figure 14. Expensive shoe materials, such as *calfskin* and *diamond*, are located on the top-left corner in the plot, while cheaper shoe products such as shoes for going to the *beach* are located on the opposite end. However, embeddings created by word2vec cannot showcase such difference.

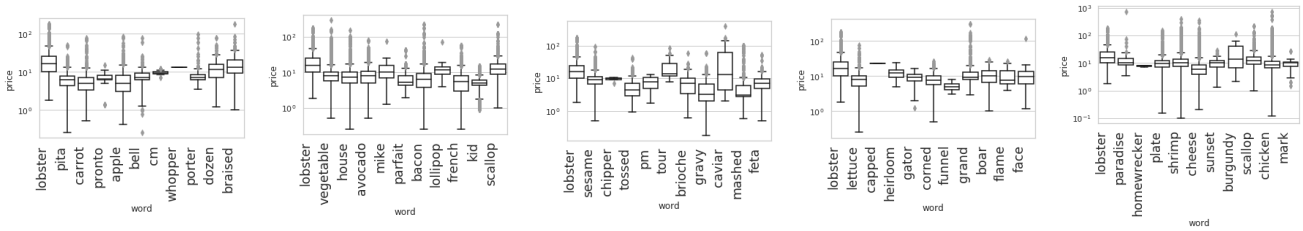
[5] https://github.com/yusanlin/word-worth-embedding

[6] To speed up the process of finding nearest neighbors in high-dimensional space, we construct KDTree of each sets of embeddings.



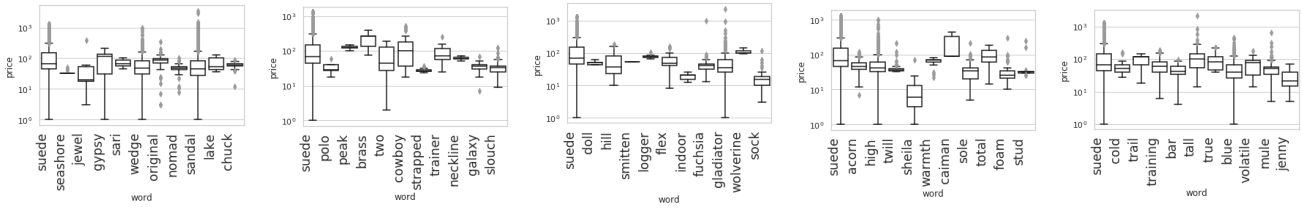
(a) Word2vec (b) WP (c) MWP (d) MWCP

Fig. 8. Baseline models



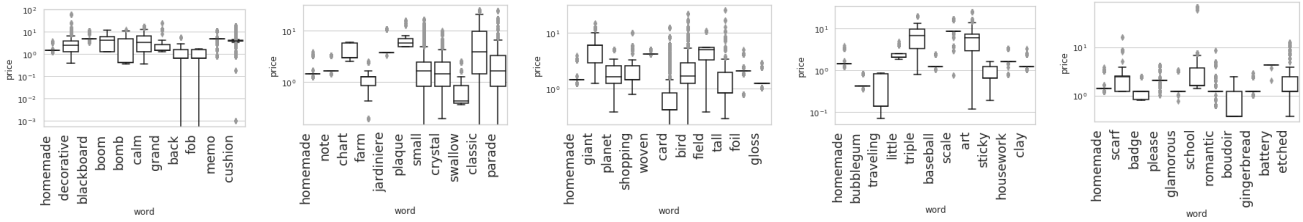
(a) Word2vec (b) WP (c) MWP (d) MWCP (e) WW

Fig. 9. Embedding comparison on menu dataset



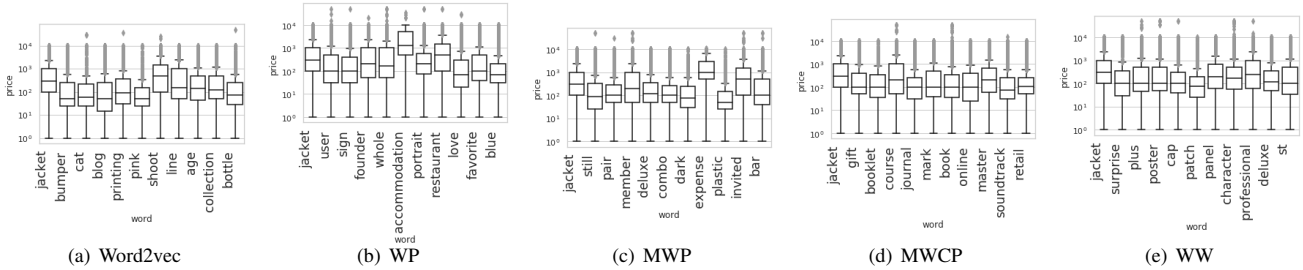
(a) Word2vec (b) WP (c) MWP (d) MWCP (e) WW

Fig. 10. Embedding comparison on shoe dataset



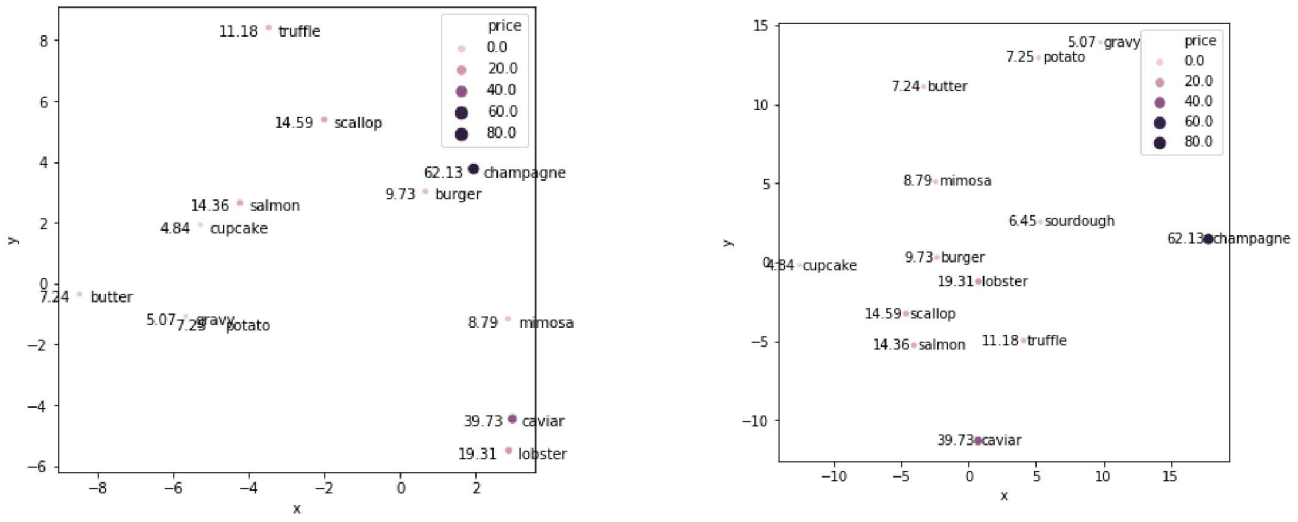
(a) Word2vec (b) WP (c) MWP (d) MWCP (e) WW

Fig. 11. Embedding comparison on retail dataset



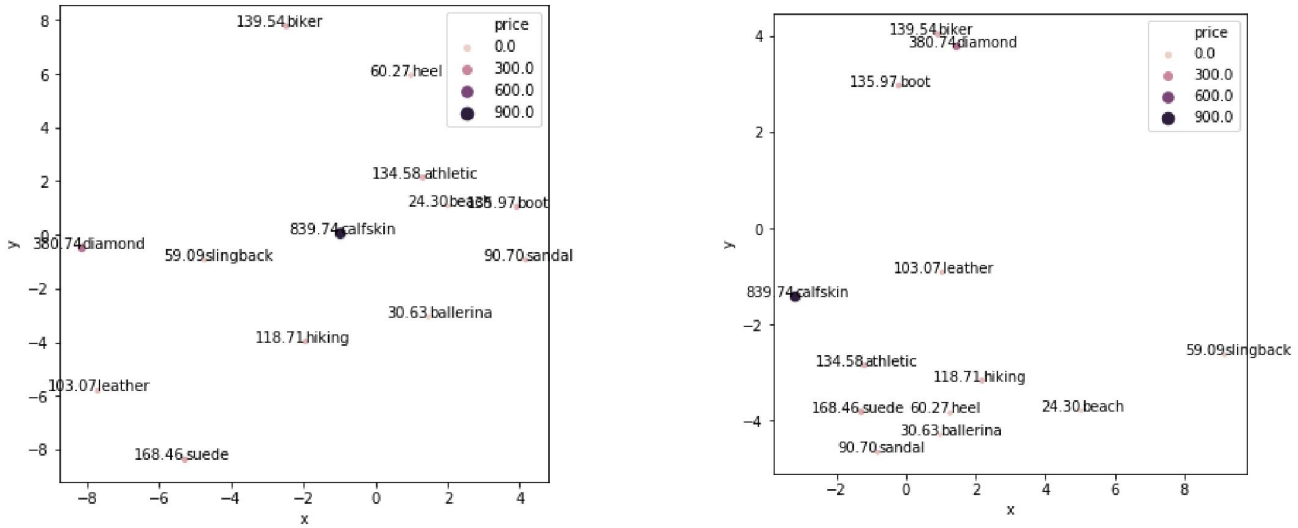
(a) Word2vec (b) WP (c) MWP (d) MWCP (e) WW

Fig. 12. Embedding comparison on reward dataset



(a) Word2vec (b) WW

Fig. 13. t-SNE-transformed embedding comparison on menu dataset



(a) Word2vec (b) WW

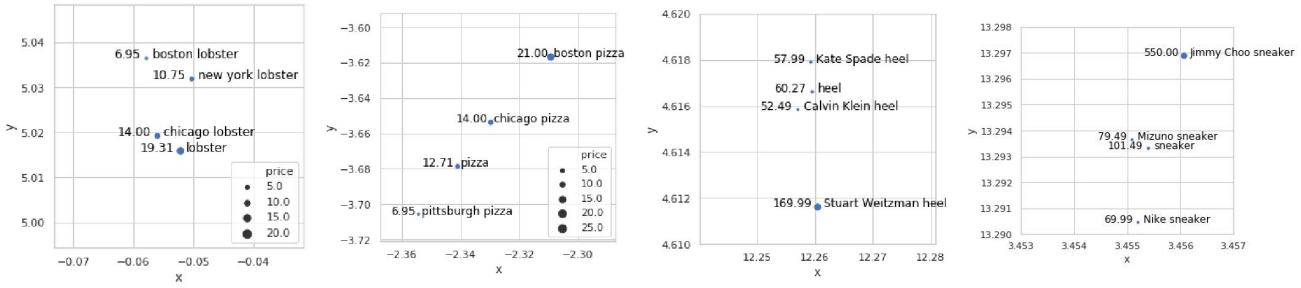
Fig. 14. t-SNE-transformed embedding comparison on shoe dataset

D. Analyses on Contextual Bias Embedding

As described previously in Section IV, we design the contextual bias that is meant to capture the effect of different contextual factors on the overall worth of a given textual description. The contextual bias is purposely designed to have the same number of dimensions as the word embeddings, in order to serve as the element-wise shift. We investigate the quality of the learned contextual biases on both the menu and shoe datasets, where the contextual factor for menu is the city of the restaurant, and the contextual factor for shoe is the brand of the shoe.

In Figure 15(a), we show how the lobster dishes are priced in different cities. On average, dishes including lobster is priced at 1931. As shown, since Boston is famously known for its locally-sourced fresh lobster (hence lower price), when adding the contextual bias of Boston given lobster, the embedding shifts significantly away from the lobster embedding, compared to lobster in other cities, such as New York and Chicago. Similar correlation between embedding position and price can be observed when we visualize the prices of pizzas in different cities, as shown in Figure 15(b).

Besides cities having an effect on price, as one can imagine,



(a) Lobster (b) Pizza (c) Heel (d) Sneaker

Fig. 15. Contextual bias visualization

TABLE IV
PRICE PREDICTION PERFORMANCES IN MSE (10E-6)

dataset	wp	word2vec	mwp	mwcp	ww
menu	36.3050	36.4088	36.1759	36.3491	35.5020
retail	8.1709	8.1690	8.1688	8.1684	8.1681
reward	5710.3388	5944.6016	5849.7515	5844.8580	5528.4521
shoe	1662.9549	1648.5325	1614.7071	1661.3844	1676.5710

brands of products can also be major factors that affect the products' prices. Stuart Weitzman is known for its luxurious boots and heels, compared to other more affordable brands such as Kate Spade. One can clearly see such shifts in the embedding space in Figure 15(c). Similarly, for sneakers, those by Jimmy Choo are for sure way more expensive than those by Nike, and one can clearly see their distance in the embedding space in 15(d).

E. Price Prediction

For price prediction, given a complete description of an object, we sum together the embedding of each word in the description, and pass the summed embedding to a linear regression model to predict price. We split each dataset to 70% training and 30% testing, and report the mean squared error (MSE) in Table IV. We intentionally choose a simple model to see whether the embedding can help with the prediction. As one can see, the embeddings obtained from WWM lead to slightly better results in the menu and reward dataset. It performs poorly in the shoe dataset, which we believe due to the nature of the shoe products, their prices are heavily dependent on the contextual factor, i.e., brands, as previously shown in 15(c) and 15(d).

F. Word Prediction

Another prediction task we conduct is the word prediction task. For each dataset, we construct the experiment data. We scan through all of the records in the dataset, and for each record, we randomly pick one word as the missing word, and the rest to be the query. For the same query, it can associates with numerous missing words, which are the target that we want to predict.

Given a query, we sum together the embedding of each word in the query, and we find the top 1000 words that have their embedding closest to the query embedding in the embedding space. We rank these words increasingly based on their distance to the query embedding, and measure the performance based on precision@k and recall@k. The results are shown in Figures 16 and 17. As can be seen, the performances of the WWM embeddings are superior than other baselines in three out of the four datasets.

VI. CONCLUSION

In this paper, through data analyses on real-world dataset, we identify the importance of understanding the economic worth of words, and propose a joint-task neural network model, Word Worth Model (WWM), that learns the word embedding capturing its underlying semantics and economic-worth. The model takes into account the two characteristics of word worths: multi-dimensionality and context dependency,. We also propose two sampling methods to prepare inputs for training WWM. Compared with the state-of-the-art model, word2vec, and other neural-network-based baseline models, embeddings produced by WWM is shown to be superior both quantitatively and qualitatively.

In the future, we plan to explore more potential architecture design of the WWM model, such as leverage more sophisticated form of sequence encoding techniques. We hope this work opens up a new line of research in learning the word embedding that considers the economic worth of words.

ACKNOWLEDGMENT

This work is supported by the National Science Foundation under Grant No. IIS-1717084.

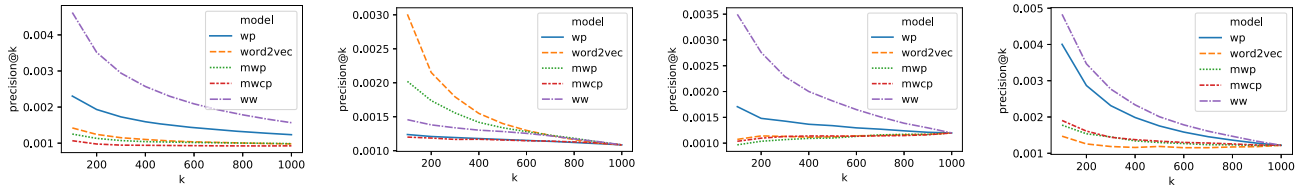
REFERENCES

[1] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

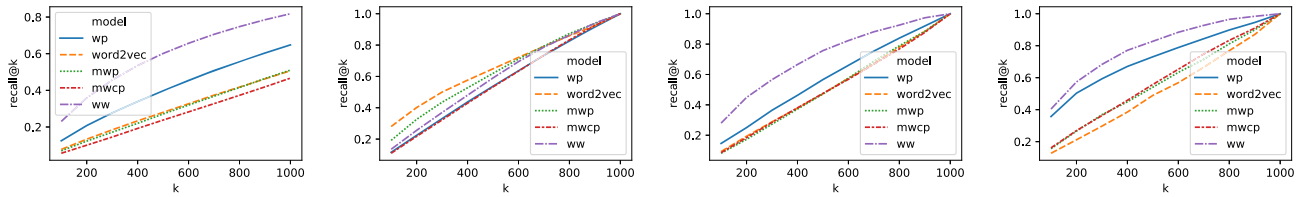
[3] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, 2013, pp. 3111–3119.

[4] A. Mnih and G. E. Hinton, "A scalable hierarchical distributed language model," in *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*, 2008, pp. 1081–1088.



(a) Menu (b) Shoe (c) Retail (d) Reward

Fig. 16. Precisions of word prediction performances



(a) Menu (b) Shoe (c) Retail (d) Reward

Fig. 17. Recalls of word prediction performances

[5] J. Reisinger and R. J. Mooney, "Multi-prototype vector-space models of word meaning," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 109–117.

[6] E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng, "Improving word representations via global context and multiple word prototypes," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 2012, pp. 873–882.

[7] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," *arXiv preprint arXiv:1503.00075*, 2015.

[8] Z. Jiang, L. Li, D. Huang, and L. Jin, "Training word embeddings for deep learning in biomedical text mining tasks," in *Bioinformatics and Biomedicine (BIBM), 2015 IEEE International Conference on*. IEEE, 2015, pp. 625–628.

[9] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus, and R. Socher, "Ask me anything: Dynamic memory networks for natural language processing," in *International Conference on Machine Learning*, 2016, pp. 1378–1387.

[10] K. Lee, L. He, M. Lewis, and L. Zettlemoyer, "End-to-end neural coreference resolution," *arXiv preprint arXiv:1707.07045*, 2017.

[11] L. He, K. Lee, M. Lewis, and L. Zettlemoyer, "Deep semantic role labeling: What works and what's next," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2017, pp. 473–483.

[12] S. Kuehling, "A fat sow named skulfi:'expensive'words in dobu island society," *Tides of Innovation in Oceania: Value, materiality and place*, p. 193, 2017.

[13] J. C. Sweeney and G. N. Soutar, "Consumer perceived value: The development of a multiple item scale," *Journal of retailing*, vol. 77, no. 2, pp. 203–220, 2001.

[14] V. Chahuneau, K. Gimpel, B. R. Routledge, L. Scherlis, and N. A. Smith, "Word salad: Relating food prices and descriptions," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, 2012, pp. 1357–1367.

[15] A. Eggert and W. Ulaga, "Customer perceived value: a substitute for satisfaction in business markets?" *Journal of Business & industrial marketing*, vol. 17, no. 2/3, pp. 107–118, 2002.

[16] Z. Yang and R. T. Peterson, "Customer perceived value, satisfaction, and loyalty: The role of switching costs," *Psychology & Marketing*, vol. 21, no. 10, pp. 799–822, 2004.

[17] Y. Bengio, A. C. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, 2013. [Online]. Available: https://doi.org/10.1109/TPAMI.2013.50

[18] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, 2014, pp. 1188–1196. [Online]. Available: http://jmlr.org/proceedings/papers/v32/le14.html

[19] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, 2016, pp. 855–864. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939754

[20] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, 2014, pp. 701–710. [Online]. Available: http://doi.acm.org/10.1145/2623330.2623732

[21] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, 2015, pp. 1067–1077. [Online]. Available: http://doi.acm.org/10.1145/2736277.2741093

[22] T. Fu, W. Lee, and Z. Lei, "Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*, 2017, pp. 1797–1806. [Online]. Available: http://doi.acm.org/10.1145/3132847.3132953

[23] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, 2017, pp. 135–144. [Online]. Available: http://doi.acm.org/10.1145/3097983.3098036

[24] M. Li, Q. Lu, Y. Long, and L. Gui, "Inferring affective meanings of words from word embedding," *IEEE Transactions on Affective Computing*, 2017.