# Field Embedding: A Unified Grain-Based Framework for Word Representation

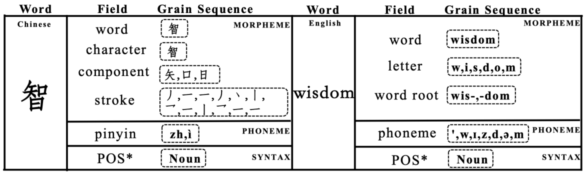
**Junjie Luo**[1,2*], **Xi Chen**[1*†], **Jichao Sun**[1*], **Yuejia Xiang**[1],
**Ningyu Zhang**[3], **Xiang Wan**[4]
[1]Tencent Jarvis Lab [2]University of Maryland
[3]Zhejiang University [4]Shenzhen Research Institute of Big Data
jjluo@umd.edu zhangningyu@zju.edu.cn wanxiang@sribd.cn
{jasonxchen,jichaosun,yuejiaxiang}@tencent.com

## Abstract

Word representations empowered with additional linguistic information have been widely studied and proved to outperform traditional embeddings. Current methods mainly focus on learning embeddings for words while embeddings of linguistic information (referred to as grain embeddings) are discarded after the learning. This work proposes a framework *field embedding* to jointly learn both word and grain embeddings by incorporating morphological, phonetic, and syntactical linguistic fields. The framework leverages an innovative fine-grained pipeline which integrates multiple linguistic fields and produces high-quality grain sequences for learning supreme word representations. A novel algorithm is also designed to learn embeddings for words and grains by capturing information that is contained within each field and that is shared across them. Experimental results of lexical tasks and downstream natural language processing tasks illustrate that our framework can learn better word embeddings and grain embeddings. Qualitative evaluations show grain embeddings effectively capture the semantic information.

## 1 Introduction

Distributed word representation, also named as word embedding, represents each word as a vector in a continuous vector space. Due to its strong ability of encoding semantic information, word embedding is useful in many downstream NLP tasks, such as text classification (Wieting et al., 2016; Yin et al., 2016), named entity recognition (NER) (Collobert et al., 2011; Sun et al., 2015), etc. Classic approaches mainly treated words as atomic tokens, such as WordVec (Mikolov et al., 2013b,a) and GloVe (Pennington et al., 2014). Recently, many researchers introduced subword information to learn advanced word embeddings for different languages, including English (Bojanowski et al., 2017) and Chinese (Yu et al., 2017; Cao et al., 2018). In this paper, we refer to subword types as linguistic fields and symbols representing subwords as grains. For example, the field *letter* grain sequence of word *wisdom* is [w, i, s, d, o, m]. Elements in the sequence are *letter* grains which are from the *letter* vocabulary, i.e., the alphabet table. Fig. 1 shows more examples of linguistic fields and grains.



| Word | Field | Grain Sequence | | Word | Field | Grain Sequence | |
|---|---|---|---|---|---|---|---|
| Chinese<br>智 | word | 智 | MORPHEME | English<br>wisdom | word | wisdom | MORPHEME |
| | character | 智 | | | letter | w,i,s,d,o,m | |
| | component | 矢,口,日 | | | word root | wis-,-dom | |
| | stroke | 丿,一,一,丿,丶,丨,𠃍,一,丨,𠃍,一,一 | | | | | |
| | pinyin | zh,ì | PHONEME | | phoneme | ',w,ɪ,z,d,ə,m | PHONEME |
| | POS* | Noun | SYNTAX | | POS* | Noun | SYNTAX |

Figure 1: Field and grain sequence examples for 智 (wisdom) and *wisdom*. * indicates the hyperfield.

However, though huge progress has been achieved, there are many challenges or limitations for fully exploiting linguistic fields' potential on learning advanced embeddings. The first challenge is producing semantically meaningful representations for input words. Such representations rely on (a) broad linguistic fields and (b) high-quality grain sequences. For fields, only morphological fields were studied, such as *letter* in English (Bojanowski et al., 2017), and *component* in Chinese (Yu et al., 2017). However, linguistics studies revealed that phonetic and syntactical fields contain rich semantic information (Beaver et al., 2007), whose utility was not fully studied before. For grain sequences, current methods only produced coarse grain sequences, whose grains seldom carry information associated with the original word. For example, grains from [w, i, s, d, o, m], the *letter* grain sequence of word *wisdom*, are simple and less meaningful letters.

Second, a customized algorithm is required to

[*] The first three authors contributed equally.

[†] Xi Chen is the corresponding author.

*Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1754–1762
June 6–11, 2021. ©2021 Association for Computational Linguistics

model both the uniqueness of each field and the relationship among fields, given the increasingly available linguistic fields as the new information. Besides their uniqueness, fields have strong correlations with each other. We can find many morpheme-syntax pairs, such as -tion:Noun and -ious:Adj and morpheme-phoneme pairs: sounds of 妈 (mother) and 码 (code), which are derived from the pair 马 (horse):ma. However, past methods might fail to capture holistic information carried across fields (Chen et al., 2015b; Cao et al., 2018). They simply put all fields together and immigrated classical word2vec algorithms, which ignored such inter-field information.

Furthermore, the value of grain embeddings on NLP tasks has not been comprehensively evaluated. Similar to word embeddings, we introduce grain embeddings which represent each grain with a semantic vector. Past work focused on word embeddings but paid little attention to learning and evaluating grain embeddings for linguistic fields. Whether grain embeddings can convey semantics and benefit NLP tasks is still not systematically studied.

To solve the above challenges, we propose a *field embedding* framework to jointly learn word and grain embeddings simultaneously. It can flexibly integrate any combination of linguistic fields. Our contributions are follows:

(1) A fine-grained pipeline (a) takes any combination of various linguistic fields, including morpheme, phoneme, and syntax, as the input, and (b) includes n-gram and grain dropping to generate high-quality grain sequences as semantically meaningful and complete representations for input words.

(2) A novel algorithm is proposed with the motivation of ubiquitous linguistic phenomena. Its loss function generates two kinds of gradients to model information contained within each field and that shared across multiple fields separately. This brings holistic information to improve the embedding quality.

(3) Extensive experimental results illustrate that our framework yields supreme word and grain embeddings in various NLP tasks. Our framework learns better word embeddings than previous methods in both lexical tasks and downstream tasks. Moreover, our learned grain embeddings outperform word embeddings in downstream tasks, such as text classification and NER. Furthermore, qualitative evaluations show that grain embeddings can effectively capture semantic information.

It is the first toolkit that can measure the effectiveness of various fields and their combinations in learning word and grain embeddings. Its simplicity and compatibility spare the laborious and time-consuming developments and evaluations of new multi-field models. The code and data will be released on Github.

## 2 Background and Related Works

**Field and Grain** A sentence is a list of words. A field describes one linguistic aspect of words. For example, in Chinese, *component* and *stroke* are morphological fields which describe the word shape, *pinyin*, a phonetic field, describes pronunciations. These are subfields, which are determined exclusively by the word. In contrast, hyperfields refer to the linguistic fields determined by both the word and its context, such as *part-of-speech (POS)*, a syntactical field. A simple and efficient way to represent field information is using sequences of symbols. Symbols are referred to as grains and sequences as grain sequences, such as *wisdom*'s *word root* grain sequence is [wis-, -dom].

**Static Embeddings** Static embeddings present each word (or grain) with a semantic vector independent of its contexts. This work focuses on exploiting linguistic potential for learning static word and grain embeddings. Past works achieved huge progress in introducing linguistic fields for advanced static word embeddings. For English, one typical linguistic field is *letter*, which has been exploited to improve word embeddings (Bojanowski et al., 2017). For Chinese, subword fields, including *character*, *component*, and *stroke*, convey fruitful semantic information (Wieting et al., 2015; Liu et al., 2017) and have been studied by CWE (Chen et al., 2015a), JWE (Yu et al., 2017), and cw2vec (Cao et al., 2018) separately. The above methods adopted shallow but efficient structures. In contrast, many methods introduced deep neural networks in learning static embeddings (Kim et al., 2018; Cao and Lu, 2017). However, they cost huge computational resources and yielded limited improvements. To keep the framework straightforward and efficient, this work adopts a shallow structure.

**Dynamic Embeddings** Dynamic embeddings are trained as deep language models and represent

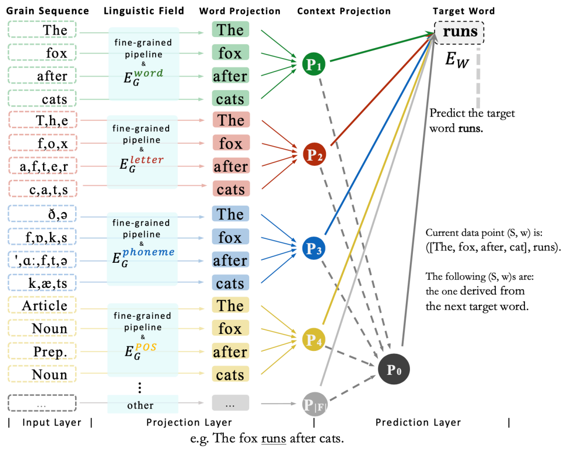

Grain Sequence
Linguistic Field
Word Projection
Context Projection
Target Word
The
fox
after
cats
T,h,e
f,o,x
a,f,t,e,r
c,a,t,s
ð,ə
f,ɒ,k,s
',ɑ:,f,t,ə
k,æ,ts
Article
Noun
Prep.
Noun
...
fine-grained pipeline & $E_G^{word}$
fine-grained pipeline & $E_G^{letter}$
fine-grained pipeline & $E_G^{phoneme}$
fine-grained pipeline & $E_G^{POS}$
⋮
other
The
fox
after
cats
...
P1
P2
P3
P4
P|F|
P0
runs
$E_W$
Predict the target word runs.
Current data point (S, w) is: ([The, fox, after, cat], runs).
The following (S, w)s are: the one derived from the next target word.
Input Layer
Projection Layer
Prediction Layer
e.g. The fox runs after cats.

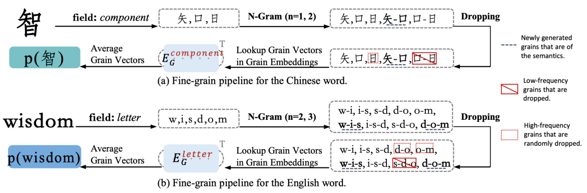
Figure 2: Field embedding cbow structure.

a whole sentence with contextual information. The dynamic embeddings, such as ELMo (Peters et al., 2018), and BERT (Devlin et al., 2018), achieved state-of-the-art performances in many NLP tasks. This work compares our grain embeddings with them on downstream tasks. However, we did not evaluate them with lexical tasks, as they cannot produce word vector without contexts directly.

## 3 Model Description

We use $C$ to denote the training corpus, $V$ the word vocabulary, and $F$ the set of selected linguistic fields. For a field $f \in F$, we generate its n-gram grain vocabulary $V_f$ by scanning the words in the whole corpus. $G_f(w) = [g_1, g_2, \dots, g_n]$ is the grain sequence of word $w$ in field $f$. Data points fed into embedding models are pairs of a target word $w_t$ and its context word set $S(w_t)$, or $S$ for simplicity. Take the sentence *The fox runs after cats* as an example and suppose *runs* is the target word. By applying *cbow*, the data points $(S, w_t)$ will be ([The, fox, after, cats], runs). By applying *skip-gram*, there are four $(S, w_t)$s: ([*ctx*], runs) for *ctx* in [The, fox, after, cats].

The *cbow* structure is shown in Fig. 2. It contains three layers: the input, projection, and prediction layer. For a data point $(S, w_t)$, multi-field inputs of $S$ in the input layer are fed to the projection layer and become projection vectors: $P_0$ to $P_{|F|}$. Each projection enters the prediction layer to predict $w_t$ and get a prediction loss. The summation of these losses is the total loss. Model's parameters include several grain embedding matrices in the projection layer and a word embedding matrix in the prediction layer. We represent word embeddings as $E_W$ of size $|V| \times d$, where $d$ is the embedding size. For each field $f$, its grain embeddings $E_G^f$ is of size $|V_f| \times d$. Word embeddings $E_W$ vectorize the target word $w_t$ while grain embeddings $E_G^f$ only vectorize grains from context words in $S(w_t)$.


智
field: component
矢,口,日
N-Gram (n=1, 2)
矢,口,日,矢-口,口-日
Dropping
矢,口,日,矢-口,口-日
Lookup Grain Vectors in Grain Embeddings
$E_G^{component}$
T
Average Grain Vectors
p(智)
wisdom
field: letter
w,i,s,d,o,m
N-Gram (n=2, 3)
w-i, i-s, s-d, d-o, o-m, w-i-s, i-s-d, s-d-o, d-o-m
Dropping
w-i, i-s, s-d, d-o, o-m, w-i-s, i-s-d, s-d-o, d-o-m
Lookup Grain Vectors in Grain Embeddings
$E_G^{letter}$
T
Average Grain Vectors
p(wisdom)
Newly generated grains that are of the semantics.
Low-frequency grains that are dropped.
High-frequency grains that are randomly dropped.

(a) Fine-grain pipeline for the Chinese word.

(b) Fine-grain pipeline for the English word.

Figure 3: Fine-grained pipeline for extracting high-quality grain sequences and projections.

### 3.1 Fine-Grained Pipeline

In the projection layer, we design a fine-grained pipeline that consists of n-gram and grain dropping to produce high-quality grain sequences. Fig. 3 shows its mechanism.

**N-Gram** Compared to the word vocabulary $V$, sizes of $V_f$ and $E_G^f$ are small, which may lead to the underfitting problem. For each field $f$, we generate n-gram grains to enlarge its grain vocabulary $V_f$. By increasing the grain vocabulary size, it enlarges each word's grain sequence for a higher capacity of carrying linguistic knowledge. As an instance, the word 智 (wisdom) contains *component* 矢 (arrow), 口 (mouth), and 日 (day). These components are not relevant to the semantics of 智 (wisdom). Without n-gram grains, $G_f(w)$ is a short sequence which hardly catches enough information, whereas n-gram grains introduce more relevant grains which carry rich semantic information. For example, after including 2-gram grains, we have 矢口 and 口日. The new grain 矢口 can be regarded as 知 (knowledge), whose semantics is similar to 智 (wisdom).

**Grain Dropping** While n-gram introduces meaningful grains, it generates many low-frequency and meaningless grains. For example, 口日 in 智 carries almost no information and seldom appears in the corpus. We filter out such noise by dropping extremely low-frequency grains. This can improve the quality of training data, reduce model parameters, and thus accelerate the training process. Moreover, motivated by dropout (Srivastava et al., 2014) and subsampling (Mikolov et al., 2013b), during the training phase, we randomly drop some high-frequency grains. At the same time, this accelerates the training process.

As shown in Fig. 3, after the pipeline process, the word 智's coarse *component* grain sequence [矢, 口, 日] is updated to [矢, 日, 矢口], which carries semantically meaningful new grains for the word 智. The case of English word *wisdom* is also the same. In short, n-gram and grain dropping help generate a better grain sequence $G_f(w)$, which will be proved to be crucial to enhance the quality of embeddings. Afterwards, we can get projection vectors. For field $f$, we can represent a word $w$ by averaging all of its grain vectors to get its word projection vector: $p_f(w) = \frac{1}{|G_f(w)|}\sum_{g\in G_f(w)} E_G^f(g)$, where $E_G^f(g)$ indicates the vector of $g$ from $E_G^f$. Then, we calculate the context projection vector $P_f(S)$:

$$P_f(S) = \frac{1}{|S|}\sum_{w\in S}\frac{1}{|G_f(w)|}\sum_{g\in G_f(w)} E_G^f(g).$$

### 3.2 Loss Function

In the prediction layer, our proposed customized algorithm contains a novel loss function which is motivated by the following linguistic phenomena. First, one field represents a linguistic attribute, such as morpheme describing shape while phoneme indicating sound. Therefore, each field contains its corresponding unique linguistic information. Moreover, fields have strong connections with each other. We can easily find many morpheme-syntax pairs, such as -tion:Noun and -ious:Adj. The morpheme-phoneme pairs are also ubiquitous: sounds of 伸 (sketch) and 绅 (sir) are from the pair 申 (apply):shen. To better model the above phenomena, we design a novel loss function to learn linguistic information contained within each field and shared across multiple fields. Next, we will show the loss function design and how it learns the information.

After calculating field projections $P_f(S)$ for all $f \in F$, we obtain the content projection of $S$ that defined as: $P_0(S) = \frac{1}{|F|}\sum_{f\in F} P_f(S)$. The objective of this problem is to minimize the negative log-likelihoods of the conditional predictive probability for a target word $w_t$ with its context word set $S(w_t)$:

$$L_{fe}(w_t) = \phi(w_t|P_0(S)) + \sum_{f\in F}\phi(w_t|P_f(S)) \quad (1)$$

The negative log-likelihood of conditional probability is defined by the negative logarithm of softmax function.

$$\phi(w_t|P) = -log\frac{exp(P^T E_W(w_t))}{\sum_{j=1}^{|V|} exp(P^T E_W(w_j))} \quad (2)$$

where $P$ represents the corresponding context projection. In practice, we adopt an optimization method based on the negative sampling and the standard gradient descend. Negative sampling is to replace the expensive denominator in Eq. (2) with a set of negative sampled words based on a frequency distribution.

$$\begin{aligned}\phi(w_t|P) = -[&\log\sigma(P^T E_W(w_t)) + \\ &\lambda\mathbb{E}_{w_{ng}\sim\mathcal{D}}\log\sigma(-P^T E_W(w_{ng})]\end{aligned} \quad (3)$$

where $\sigma$ is a sigmoid function, $\lambda$ is the number of negative samples, $\mathbb{E}_{w_{ng}\sim\mathcal{D}}[\cdot]$ represents expectation, and the negative sampled word $w_{ng}$ belongs to word frequency distribution $\mathcal{D}$. Given a specific corpus $C$, the objective likelihood is $\mathcal{L}(C) = \sum_{w_t\in C} L_{fe}(w_t)$.

Our loss function $L_{fe}$ in Eq. (1) contains two terms. For the first term, the gradient of $P_0$, $grad_0$, is back propagated to every grain across whole fields. This gradient $grad_0$ can be interpreted as updated linguistic information sharing by all the fields. For the second term, the gradient of field projection $P_f$, $grad_f$, only updates the grain vectors within the specific field $f$. This gradient $grad_f$ can be interpreted as the unique linguistic information of field $f$. In this case, the gradient that updates each grain in field $f$ is $grad_f + \frac{1}{|F|}grad_0$, which contains both unique and shared linguistic information. Existing methods mainly used only (a) the first term, such as the usage of $L_{w2v} = f(w_t|P_0)$ in word2vec, or (b) the second term, $L_{jwe} = \sum_{f\in F} f(w_t|P_f)$ in JWE. These methods chose either shared or field-specific information, which might be not complete enough for learning embeddings.

## 4 Training and Evaluations

### 4.1 Training Setups

**Training Corpus** We adopt the benchmark corpus, both Chinese and English Wikipedia data, to train embeddings with our framework. For Chinese, the segmentation tool is *jieba*, which was widely used in Chinese NLP works (Li et al., 2019). We set the minimal word frequency as 10, obtaining 390,106 unique words. We set the n-grams of *character* and *POS* as 1 and others as 4. The first 10,000 grains ordered by frequency are kept. For English corpus, we set the minimal word frequency as 30, which yields 649,068 unique words. We extend *letter* and *phoneme* to 3-gram, and set *POS* to 1-gram. First 20,000 common grains are kept.

**Baselines** To assess the effectiveness of our framework, we compare it with several state-of-the-art algorithms. **Word2vec** only uses the *word* itself as a field, and is an effective and efficient toolkit for learning word embedding. It adopts the $L_{w2v}$ to train the embeddings. **CWE** combines both *word* and *character* as fields and trains embeddings in a *cbow* structure. It also adopts the $L_{w2v}$. **JWE** incorporates *word*, *character*, and *component* with the *cbow* structure and adopts the $L_{jwe}$. **cw2vec** uses n-gram *stroke* to train embeddings in a *skip-gram* method and adopts the $L_{w2v}$.

**Hyperparameters** For a fair comparison, each word and grain embedding is of 200 dimensions for all algorithms. We set the window size and iteration to 5, the initial learning rate to 0.025, and the negative samples to 10.

### 4.2 Evaluation Tasks

In the task evaluation part, lexical tasks are conducted to evaluate word embeddings and the effectiveness of new linguistic fields, fine-grained pipeline, and our novel loss function. Downstream tasks are conducted to evaluate performances of grain embeddings compared to word embeddings. Qualitative analysis is used to validate semantic information in grain embeddings. All task datasets are widely used in previous word embedding works.

**Lexical Evaluations** Lexical evaluations include word similarity and word analogy, which are widely applied to evaluate the quality of word embeddings.

(a) *Word Similarity* This task evaluates the model's ability of capturing the semantic relevance between given word pairs. We adopt datasets Sim240 and Sim297 from (Chen et al., 2015b) for word similarity tasks to evaluate Chinese word embeddings, and use Sim353 from (Mikolov et al., 2013a) for English word embeddings. For each pair of words in each dataset, a human-labeled score is provided. We compute the cosine similarity of each word pair and use the Spearman correlation to measure the quality of word embeddings.

(b) *Word Analogy* This task evaluates whether the word embeddings capture the linguistic relationship between word pairs. Given three words like Berlin, Germany, Paris, the model should infer that the most similar word vector vec(France) with vec(Germany)-vec(Berlin)+vec(Paris). We adopt the Chinese dataset provided by (Chen et al., 2015b), and English datasets from Google (Mikolov et al., 2013a) and MSR (Mikolov et al., 2013c).

**Downstream Task** In downstream tasks, we use our embeddings to represent words in a text or sentence as input features. Both word embeddings and grain embeddings can construct a word representation for a word $w$. By word embeddings $E_W$, a word $w$ can be represented as $E_W(w)$. By grain embeddings, the representation of word $w$ can be constructed by concatenating all word projection $p_f(w)$ together, where $f \in F$ are available fields. In downstream tasks, the learned embeddings are frozen and not updated in training phases.

(a) *Text Classification* We follow cw2vec and select five topics in the Chinese dateset FudanCorpus and obtain 5,885 texts. For English, we use NewsGroup and obtain 20 topics as well as 18,756 texts. We average word representation of words in a text as its input feature vectors. We build the classifier by using SVM in *sklearn* and utilize five-fold cross validation to obtain accuracy scores.

(b) *Named Entity Recognition* For Chinese, we adopt Boson, which contains 19,214 sentences and five entity categories and randomly separate it into train, validation, and test parts by 8:1:1. For English, we use CoNLL2003, which contains 16,477 sentences, 4 entity categories, and its own dataset segmentation. We develop a CRF model (Lafferty et al., 2001) based on PyTorch as the classifier. We adopt a simple Embed-CRF structure to evaluate the embedding quality and a complicated BiLSTM-CRF to validate both static and dynamic embeddings.

**Qualitative Evaluation** It is based on the vectors of selected *character*, *component* and *pinyin* grains from learned field grain embeddings. We evaluate their top similar words from the learned embeddings, which are retrieved based on the cosine similarity.

## 5 Experimental Result

The fields and their logograms we used in this work are: *word* W, *POS* Pos; if Chinese, *character* H, *component* C, *stroke* S, *pinyin* P; if English, *letter* C, *phoneme* P.

| | Method | Sim240 | Sim297 | Analogy |
|---|---|---|---|---|
| Chinese | word2vec | 48.16 | 58.03 | 72.4 |
| | CWE | 50.8 | 55.33 | 33.37 |
| | JWE | 53.44 | 58.95 | 66.4 |
| | cw2vec | 49.88 | 48.42 | 39.35 |
| | W.Pos | **58.08** | 59.24 | 78.8 |
| | W.C.Pos | 56.66 | 59.6 | **79.43** |
| | W.C.P.Pos | 56.53 | **60.88** | 79.12 |
| | **Method** | **Sim353** | **AnaMSR** | **AnaGoogle** |
| English | word2vec | 65.14 | 57.46 | 65.12 |
| | CWE | 65.47 | 57.35 | 64.81 |
| | W.C | 68.38 | **62.16** | **70.09** |
| | W.P.Pos | 68.32 | 61.51 | 70.02 |
| | W.C.P.Pos | **68.51** | 59.01 | 69.5 |

Table 1: Performance on lexical tasks achieved by word embeddings generated by the proposed method and state-of-the-art methods. Spearman correlation coefficient is presented in percentage (%).

## 5.1 Lexical Evaluation

The lexical tasks evaluate word embeddings in terms of different field combinations, using pipeline or not, and loss functions to verify the effectiveness of our proposed methods. Results in Table 1 show that our proposed framework achieves best performances for different tasks. For instance, in Chinese Sim240 task, our model W.Pos gets the best similarity score 58.08%, with a 4.64% increase from best baseline method JWE. Next, we analyze sources of the improvements.

**Linguistic Field** In Table 1, for both Chinese and English, our models which integrate phoneme and syntax fields perform better on three different tasks and outperform other existing models. For example, W.C.P.Pos achieves best performances in both Chinese Sim297 (1.93% increase from the best baseline JWE) and English Sim353 (3.04% increase from the best baseline CWE). This supports phoneme and syntax fields carry new linguistic fields which previously widely used morphological fields do not contain. Moreover, putting all fields together to train embeddings does not guarantee that the learned word embeddings can achieve best performances for all tasks, such as the Chinese W.C.P.Pos model in Table 1. This indicates some linguistic fields bring more noise than semantic information for the corresponding task. Instead, finding the best field combination to train embeddings for specific tasks is more important. Our proposed framework makes it easy to explore the best field combination for each specific task.

| Method | **NewsGroup** (English) | | **Fudan** (Chinese) | |
|---|---|---|---|---|
| | $E_W$ | $E_G$ | $E_W$ | $E_G$ |
| Word2Vec | 69.48 | - | 93.88 | - |
| W.C | 72.98 | 75.36 | 94.24 | 94.55 |
| W.C.P | 73.03 | 76.82 | 93.92 | 94.61 |
| W.C.P.Pos | 77.04 | **80.12** | 94.26 | **95.00** |

Table 2: Comparison of the proposed method with Word2Vec on the text classification tasks. Accuracy is presented in percentage (%).

**Fine-Grained Pipeline** Fig.4 verifies the fine-grained pipeline's effectiveness. Given fields and loss function of the model, leveraging the whole fine-grained pipeline including n-gram and grain dropping can lead to advanced word embeddings. For example, in the W.H.C model, the fine-grained pipeline yields a 9.13% increase in the Chinese Analogy task. Beyond this, Fig. 5 further validates the effectiveness of grain dropping. In the W.H.P model, grain dropping achieves a 1.57% improvements in Analogy task. The improvements demonstrate that our fine-grained pipeline does produce high-quality grain sequences and can successfully capture more linguistic information that previous works missed.

**Loss Function** Fig. 6 asserts the effectiveness of our loss function. $L_{fe}$ loss function outperforms other loss functions in all tasks. For example, in Sim240 task, $L_{fe}$ achieves 4.49% increase compared to $L_{w2v}$ and 3.04% increase compared to $L_{jwe}$. It's the same for Sim297 and Analogy. Compared with $L_{w2v}$ and $L_{jwe}$, $L_{fe}$ considers prominent linguistic phenomena such as morpheme-phoneme pairs. Therefore, it successfully captures the within-field and crossing-field information.

## 5.2 Downstream Task

We conduct both Chinese and English downstream tasks, including text classification and NER, to test the performances of *word embeddings* and *grain embeddings*.

**Text Classification** Table 5.2 shows with more linguistic fields, both word and grain embeddings gain performance improvements on text classification tasks. This reveals that phoneme and syntax fields can improve the quality of both word and grain embeddings. Moreover, grain embeddings $E_G$ always outperform word embeddings $E_W$. In NewsGroup task, $E_G$ of W.C.P.Pos exceed $E_W$ with 3.08%. Furthermore, compared with word embeddings, grain embeddings make larger improvements with more fields. For exam-

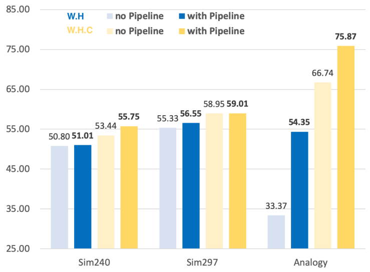


Figure 4: Fine-grained pipeline performances on lexical tasks.

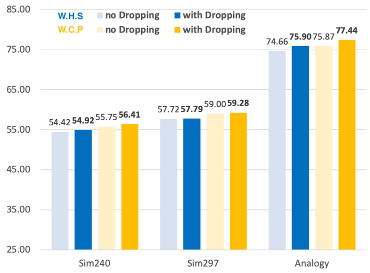


Figure 5: Grain dropping performances on lexical tasks.

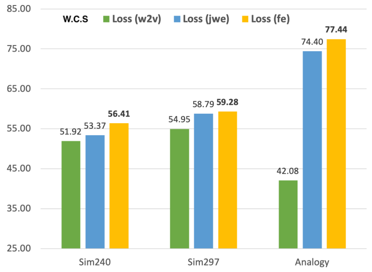


Figure 6: Loss function performances on lexical tasks.

| **Method** | **CoNLL2003** (English) | | **Boson** (Chinese) | |
|---|---|---|---|---|
| | $E_W$ | $E_G$ | $E_W$ | $E_G$ |
| Word2Vec | 77.04 | - | 64.13 | - |
| W.C | 77.19 | 80.35 | 64.96 | 65.08 |
| W.C.P | 78.06 | 82.01 | 65.08 | 65.34 |
| W.C.P.Pos | 78.44 | **86.92** | 65.40 | **71.75** |

Table 3: Comparison of the proposed method with the Embed-CRF structure and Word2Vec on NER. F1 score is presented in percentage (%).

| **Method** | **F1 Score (%)** |
|---|---|
| W.C | 91.82 |
| W.C.P | 92.16 |
| W.C.P.Pos | **92.34** |
| Word2Vec (Mikolov et al., 2013b) | 90.72 |
| ELMo (Peters et al., 2018) | 92.22 |
| BERT (Devlin et al., 2018) | **92.80** |

Table 4: Different embeddings with BiLSTM-CRF structure on the CoNLL2003 NER task.

ple, from W.C to W.C.P.Pos, $E_W$ achieve a 4.06% increase while $E_G$ achieves a 4.76% increase in NewsGroup task. Such additional improvement derives from additional linguistic information that is not included in word embeddings. This strongly indicates that grain embeddings carry more linguistic information than the associated word embeddings and can be a better alternative to word embeddings.

**Named Entity Recognition** As to NER performances of Table 3, a similar pattern is observed to that of text classification. It demonstrates that more linguistic fields benefit the NER tasks. For example, from W.C.P to W.C.P.Pos, grain embeddings gain 4.91% and 6.41% improvements in CoNLL2003 and Boson tasks. The reason for the improvements is that the hyperfield *POS* carries the part of a sentence’ s syntactical information, which is crucial in sequence labeling tasks. This illustrates the hyperfield’s significance in learning embeddings. Moreover, it shows grain embeddings outperform word embeddings. For instance, in W.C.P.Pos model, F1 scores of grain embeddings exceed that of word embeddings with 8.48% and 6.35% in CoNLL2003 and Boson tasks.

To further prove that our embeddings are effective with complicated neural networks, we adopt Embed-BiLSTM-CRF and conduct experiments on CoNLL2003. Besides static embedding Word2Vec, dynamic embedding methods, such as ELMo and BERT, are also listed as baselines for comparison, as shown in Table 4. In terms of static embeddings, additional fields still benefit the task, with around 0.54% increase per field from Word2Vec to W.C.P.Pos. This shows that, even in complex neural networks, grain embeddings are superior to word embeddings, and phoneme and syntax are useful.

For dynamic embeddings, though it is marginally inferior to BERT, our W.C.P.Pos is better than EMLo by 0.12%. It indicates our framework exploits the potential of static embeddings with multiple fields which surpasses the relatively shallow dynamic embeddings. This also suggests the potential of introducing multi-fields to dynamic embeddings. Our static embeddings also enjoy other advantages. The model’s structure is simple and straightforward and parameter size is small. It requires less corpus and resources to train compared to BERT, which is a complicated deep neural networks. In downstream tasks, though BERT outperforms our model, it bears expensive costs of model complexity and computational resources. Moreover, the dynamic embeddings cannot represent the independent word or gain, whereas our model yields high quality representations for them and achieves best performances in lexical tasks.

### 5.3 Qualitative Evaluation

We evaluate the embeddings' abilities to uncover the semantic relatedness of words, characters, and components through case studies based on a model trained with *character*, *component*, and *pinyin*. Taking 申 (apply) as an example, which is also an ancient state name and a popular last name in China, it can be a Chinese character or word. We list its closest words from word embeddings in Table 5 where we treat 申 as a character and a word. When it is a character, most of the closest words are semantically related to *apply*. When it acts as a word, the closest words related to country name and last name meanings in 申. For example, 赵 (Zhao), 殷 (Yin) are ancient state names and last names, and 定公 (Duke Ding) 晋昭公 (Duke Zhao of Jin) are dukes in ancient China. This reveals that grain embeddings can supplement the word embeddings for a more complete semantic representation.

We further take the *component* 疒 (illness) as an example and Table 5 shows its closest characters and words. All of the closest characters and words are semantically related to the *component* 疒. Most of them are related to diseases, symptoms and other medical terms, such as words 疾病 (disease), 感染 (infection) and characters 症 (symptom), 疮 (score). Most of them contain *component* 疒, but 肿 (gangrene), 患 (suffer), and 感染 (infection) without 疒 also share the similar semantics. Moreover, we study *pinyin* t-òng, sound of 痛 (pain), and list its closest words and characters in Table 5 and observe a similar phenomenon to 疒. These closest characters have similar semantic meaning with 痛 (pain), whose pinyin is t-òng, such as words 疼痛 (pain), 头痛 (headache), and characters 疮 (sore), 瘫 (paralysis). The qualitative analysis shows that our proposed models leverage both external context co-occurrence information and internal morphological and phonetic information. The medical information stored in above grain embeddings could be utilized for clinical NER tasks.

| Vector | Embedding | Top 5 Closest Results |
|---|---|---|
| $E_W$ 申 | $E_W$ | 赵 (Zhao), 殷 (Yin), 定公 (Duke Ding),<br>楚 (Chu), 晋昭公 (Duke Zhao of Jin) |
| $E_G^{char}$ 申 | $E_W$ | 申请 (application), 申请人 (applicant),<br>签证 (visa), 资格 (qualification), 入境 (entry) |
| $E_G^{comp}$ 疒 | $E_W$ | 疾病 (disease), 感染 (infection),<br>疼痛 (pain), 肺 (lung), 症状 (symptom) |
|  | $E_G^{char}$ | 症 (symptom), 疮 (sore), 癫 (epilepsy),<br>肿 (swollen), 疽 (gangrene) |
| $E_G^{pinyin}$ t-òng | $E_W$ | 痛 (pain), 流泪 (tear), 焦虑 (anxiety),<br>痛楚 (pain), 呕 (vomit) |
|  | $E_G^{char}$ | 痛 (pain), 疚 (remorse), 悸 (palpitate),<br>症 (disease), 嚏 (sneeze) |

Table 5: Qualitative analysis. For a Vector, from an Embedding top 5 closest results are listed.

## 6 Conclusion

We propose a flexible *field embedding* framework to jointly learn both word and grain embeddings by incorporating morphological, phonetic, and syntactical linguistic fields simultaneously. Our proposed framework leverages an innovative fine-grained pipeline, including n-grams and grain dropping, as well as a novel loss function to capture the information contained within each linguistic field and shared across multiple fields. By introducing phonetic and syntactical linguistic fields and leveraging our fine-grained pipeline and loss function, our framework is capable of learning better word embeddings in terms of word similarity and analogy. Furthermore, we systemically investigate the effectiveness of grain embeddings and provides the evidence that grain embeddings can be a better alternative to word embeddings for word representations. Experimental results show that grain embeddings outperform word embeddings in several downstream NLP tasks, such as text classification and named entity recognition. The qualitative analysis illustrates that grain embeddings can effectively capture semantic information.

## Acknowledgments

We would like to thank Prof. Jordan Boyd-Graber from University of Maryland and Dr. Yefeng Zheng from Tencent for their helpful suggestions. This work was supported by the Key-Area Research and Development Program of Guangdong Province [2020B0101350001].

## References

David I Beaver, Brady Clark, Edward Stanton Flemming, T Florian Jaeger, and Maria Wolters. 2007. When semantics meets phonetics: Acoustical studies of second-occurrence focus. *Language*, 83(2):245–276.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Shaosheng Cao and Wei Lu. 2017. Improving word embeddings with convolutional feature learning and subword information. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Shaosheng Cao, Wei Lu, Jun Zhou, and Xiaolong Li. 2018. cw2vec: Learning chinese word embeddings with stroke n-gram information. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Heng Chen, Junying Liang, and Haitao Liu. 2015a. How does word length evolve in written chinese? *PloS one*, 10(9):e0138567.

Xinxiong Chen, Lei Xu, Zhiyuan Liu, Maosong Sun, and Huanbo Luan. 2015b. Joint learning of character and word embeddings. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(Aug):2493–2537.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Yeachan Kim, Kang-Min Kim, Ji-Min Lee, and SangKeun Lee. 2018. Learning to generate word representations using subword information. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2551–2561.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Xiaoya Li, Yuxian Meng, Xiaofei Sun, Qinghong Han, Arianna Yuan, and Jiwei Li. 2019. Is word segmentation necessary for deep learning of chinese representations? In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 3242–3252.

Frederick Liu, Han Lu, Chieh Lo, and Graham Neubig. 2017. Learning character-level compositionality with visual features. *arXiv preprint arXiv:1704.04859*.

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

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015. Modeling mention, context and entity with neural networks for entity disambiguation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Towards universal paraphrastic sentence embeddings. *arXiv preprint arXiv:1511.08198*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Charagram: Embedding words and sentences via character n-grams. *arXiv preprint arXiv:1607.02789*.

Rongchao Yin, Quan Wang, Peng Li, Rui Li, and Bin Wang. 2016. Multi-granularity chinese word embedding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 981–986.

Jinxing Yu, Xun Jian, Hao Xin, and Yangqiu Song. 2017. Joint embeddings of chinese words, characters, and fine-grained subcharacter components. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 286–291.