Variations to the SkipGram Model

VL Embeddings

Uni Heidelberg

SS 2019
Generalisation of SkipGram to arbitrary contexts

- Neural embeddings so far:
  - linear bag-of-words context (with window size $n$)
  
  \[
  \text{Die \, \underline{kleine \, graue \, Maus} \, frißt \, \underline{den \, leckeren \, Käse}}
  \]

- What about other types of contexts?
Generalisation of SkipGram to arbitrary contexts

- Neural embeddings so far:
  - linear bag-of-words context (with window size $n$)

  Die **kleine graue Maus** frißt den leckeren Käse

- What about other types of contexts?

Levy and Goldberg (2014):
Dependency-based word embeddings
Starting point: SkipGram

• **Recap**: Skipgram with negative sampling (SGNS)
  - Each word \( w \in W \) is associated with a vector \( v_w \in \mathbb{R}^d \)
  - Each context \( c \in C \) is associated with a vector \( v_c \in \mathbb{R}^d \)
  - \( W \) is the word vocabulary
  - \( C \) is the context vocabulary
  - \( d \) is the embedding dimensionality

• Vector entries are the parameters \( \theta \) that we want to learn

• Given: dataset \( D \) of observed \((w, c)\) pairs in the corpus

• Objective: maximise the probability for seen word-context pairs \((w, c)\) in \( D \) and minimise the probability for random word-context pairs in \( D' \)
starting point: SkipGram

- **Recap:** Skipgram with negative sampling (SGNS)
  - Each word \( w \in W \) is associated with a vector \( v_w \in \mathbb{R}^d \)
  - Each context \( c \in C \) is associated with a vector \( v_c \in \mathbb{R}^d \)
  - \( W \) is the word vocabulary
  - \( C \) is the context vocabulary
  - \( d \) is the embedding dimensionality

- Vector entries are the parameters \( \theta \) that we want to learn
- Given: dataset \( D \) of observed \( (w, c) \) pairs in the corpus

- SGNS training objective:
  \[
  \arg\max_{v_w, v_c} \left( \sum_{(w, c) \in D} \log \sigma(v_c \cdot v_w) + \sum_{(w, c) \in D'} \log \sigma(-v_c \cdot v_w) \right)
  \]
  where \( \sigma(x) = 1/(1 + e^x) \) sigmoid function
Starting point: SGNS

SGNS

- Observed word-context pairs will end up with similar embeddings
- Context is defined as a bag-of-words window with size $n$
- Model is unsensitive to position in context window
Dependency-based embeddings

Starting point: SGNS

SGNS

- Observed word-context pairs will end up with similar embeddings
- Context is defined as a bag-of-words window with size $n$
- Model is unsensitive to position in context window

Dependency-based embeddings

- Replace bag-of-words context with syntactic context
Dependency-based word embeddings

Australian scientist **disCOVERs** star with telescope

- Which word-context pairs does SGNS extract for **disCOVER**?
- Which word-context pairs does SGNS extract for **star**?
Australian scientist **discovers** star with telescope

- Which word-context pairs does SGNS extract for **discover**?
- Which word-context pairs does SGNS extract for **star**?
- How does the dependency tree for this sentence look like?
- What contexts could a dependency-based model extract?
Dependency-based word embeddings

Australian scientist discovers stars with telescope
• Collapse preposition relations into single arc (attach PP obj to head of preposition but keep information on prep form).
Dependency-based word embeddings

- Collapse preposition relations into single arc (attach PP obj to head of preposition but keep information on prep form).

**Word**  **Contexts**
Dependency-based word embeddings

- Collapse preposition relations into single arc (attach PP obj to head of preposition but keep information on prep form).

**Word** | **Contexts**
---|---
Australian | scientist / amod⁻¹
Dependency-based word embeddings

Australian scientist discovers stars with telescope

- Collapse preposition relations into single arc (attach PP obj to head of preposition but keep information on prep form).

<table>
<thead>
<tr>
<th>Word</th>
<th>Contexts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australian</td>
<td>scientist / amod⁻¹</td>
</tr>
<tr>
<td>scientist</td>
<td>australian / amod, discovers / subj⁻¹</td>
</tr>
</tbody>
</table>
Dependency-based word embeddings

- Collapse preposition relations into single arc (attach PP obj to head of preposition but keep information on prep form).

<table>
<thead>
<tr>
<th>WORD</th>
<th>CONTEXTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australian</td>
<td>scientist / amod$^{-1}$</td>
</tr>
<tr>
<td>scientist</td>
<td>australian / amod, discovers / subj$^{-1}$</td>
</tr>
<tr>
<td>discovers</td>
<td>scientist / subj, star / obj, telescope / prep_with</td>
</tr>
</tbody>
</table>
Dependency-based word embeddings

- Collapse preposition relations into single arc (attach PP obj to head of preposition but keep information on prep form).

<table>
<thead>
<tr>
<th>Word</th>
<th>Contexts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australian</td>
<td>scientist / amod(^{-1})</td>
</tr>
<tr>
<td>scientist</td>
<td>australian / amod, discovers / subj(^{-1})</td>
</tr>
<tr>
<td>discovers</td>
<td>scientist / subj, star / obj, telescope / prep_with</td>
</tr>
<tr>
<td>star</td>
<td>discovers / obj(^{-1})</td>
</tr>
</tbody>
</table>
Dependency-based word embeddings

- Collapse preposition relations into single arc (attach PP obj to head of preposition but keep information on prep form).

<table>
<thead>
<tr>
<th>WORD</th>
<th>CONTEXTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australian</td>
<td>scientist / amod⁻¹</td>
</tr>
<tr>
<td>scientist</td>
<td>australian / amod, discovers / subj⁻¹</td>
</tr>
<tr>
<td>discovers</td>
<td>scientist / subj, star / obj, telescope / prep_with</td>
</tr>
<tr>
<td>star</td>
<td>discovers / obj⁻¹</td>
</tr>
<tr>
<td>telescope</td>
<td>discovers / prep_with⁻¹</td>
</tr>
</tbody>
</table>
Dependency-based word embeddings

Extract syntactic context

- Parse the corpus
- for a target word $w$ with dependents $m_1, \ldots, m_k$ and a head $h$

$\Rightarrow$ extract contexts $(m_1, lbl_1), \ldots, (m_k, lbl_k), (h, lbl_h^{-1})$
Dependency-based word embeddings

- Given the following tree in Universal Dependencies schema:

- Extract all context words for
  - schlimmer
  - Akzent
Dependency-based word embeddings

• Given the following tree in Universal Dependencies schema:

• Extract all context words for
  • schlimmer
  • Akzent

  Nix/nsubj, ist/cop, Akzent/obl, !/punct
Given the following tree in Universal Dependencies schema:

- Extract all context words for
  - schlimmer
  - Akzent

Nix/nsubj, ist/cop, Akzent/obl, !/punct
deutscher/amod, schlimmer/prep_als"
Advantages of dependency-based embeddings

- Captures context that is functionally related but far away
- Ignores words that are close by but not related
- Captures general functional relations (e.g. *stars* are objects of *discovery*, *scientists* are subjects of *discovery*)
Advantages of dependency-based embeddings

- Captures context that is functionally related but far away
- Ignores words that are close by but not related
- Captures general functional relations (e.g. *stars* are objects of *discovery*, *scientists* are subjects of *discovery*)

**Hypothesis:** Dependency-based embeddings will capture more functional and less topical similarity.
Related work

• Previous work in **distributional semantics**
  • Lin (1998)
  • Padó and Lapata (2007)
  • Baroni and Lenci (2010)
  • ...

Syntax-based semantic space models
Experiments: Settings & Data

Settings

• 3 Training conditions
  • BoW context with size $k = 5$
  • BoW context with size $k = 2$
  • Dependency context

• modified version of SkipGram implementation

• negative samples $= 15$

• embedding dimensions $= 300$

Data

• All embeddings trained on English Wikipedia
  • all tokens lower-cased
  • all word-context pairs less frequent than 100 were ignored

• Vocabulary size: 175,000 words

• Over 900,000 distinct syntactic contexts
Qualitative Evaluation

• Manually inspect 5 most similar words (cosine similarity) of a given target word

Findings:
⇒ BoW finds words that associate with \( w \)
⇒ Deps finds words that behave like \( w \)

Domain similarity vs. functional similarity
## Qualitative Evaluation

<table>
<thead>
<tr>
<th>Target Word</th>
<th>BoW5</th>
<th>BoW2</th>
<th>Deps</th>
</tr>
</thead>
<tbody>
<tr>
<td>batman</td>
<td>nightwing</td>
<td>superman</td>
<td>superman</td>
</tr>
<tr>
<td></td>
<td>aquaman</td>
<td>superboy</td>
<td>superboy</td>
</tr>
<tr>
<td></td>
<td>catwoman</td>
<td>aquaman</td>
<td>supergirl</td>
</tr>
<tr>
<td></td>
<td>superman</td>
<td>catwoman</td>
<td>catwoman</td>
</tr>
<tr>
<td></td>
<td>manhunter</td>
<td>batgirl</td>
<td>aquaman</td>
</tr>
<tr>
<td>hogwarts</td>
<td>dumbledore</td>
<td>overnight</td>
<td>sunnydale</td>
</tr>
<tr>
<td></td>
<td>hallows</td>
<td>sunnydale</td>
<td>collinwood</td>
</tr>
<tr>
<td></td>
<td>half-blood</td>
<td>garderobe</td>
<td>calarts</td>
</tr>
<tr>
<td></td>
<td>malfoy</td>
<td>blandings</td>
<td>greendale</td>
</tr>
<tr>
<td></td>
<td>snape</td>
<td>collinwood</td>
<td>millfield</td>
</tr>
<tr>
<td>turing</td>
<td>nondeterministic</td>
<td>non-deterministic</td>
<td>pauling</td>
</tr>
<tr>
<td></td>
<td>non-deterministic</td>
<td>finite-state</td>
<td>hotelling</td>
</tr>
<tr>
<td></td>
<td>computability</td>
<td>nondeterministic</td>
<td>heting</td>
</tr>
<tr>
<td></td>
<td>deterministic</td>
<td>buchi</td>
<td>lessing</td>
</tr>
<tr>
<td></td>
<td>finite-state</td>
<td>primality</td>
<td>hamming</td>
</tr>
<tr>
<td>florida</td>
<td>gainesville</td>
<td>fla</td>
<td>texas</td>
</tr>
<tr>
<td></td>
<td>fla</td>
<td>alabama</td>
<td>louisiana</td>
</tr>
<tr>
<td></td>
<td>jacksonville</td>
<td>gainesville</td>
<td>georgia</td>
</tr>
<tr>
<td></td>
<td>tampa</td>
<td>tallahassee</td>
<td>california</td>
</tr>
<tr>
<td></td>
<td>lauderdale</td>
<td>texas</td>
<td>carolina</td>
</tr>
<tr>
<td>object-oriented</td>
<td>aspect-oriented</td>
<td>aspect-oriented</td>
<td>event-driven</td>
</tr>
<tr>
<td></td>
<td>smalltalk</td>
<td>event-driven</td>
<td>domain-specific</td>
</tr>
<tr>
<td></td>
<td>event-driven</td>
<td>objective-c</td>
<td>rule-based</td>
</tr>
<tr>
<td></td>
<td>prolog</td>
<td>dataflow</td>
<td>data-driven</td>
</tr>
<tr>
<td></td>
<td>domain-specific</td>
<td>4gl</td>
<td>human-centered</td>
</tr>
<tr>
<td>dancing</td>
<td>singing</td>
<td>singing</td>
<td>singing</td>
</tr>
<tr>
<td></td>
<td>dance</td>
<td>dance</td>
<td>rapping</td>
</tr>
<tr>
<td></td>
<td>dances</td>
<td>dances</td>
<td>breakdancing</td>
</tr>
<tr>
<td></td>
<td>dancers</td>
<td>breakdancing</td>
<td>miming</td>
</tr>
<tr>
<td></td>
<td>tap-dancing</td>
<td>clowning</td>
<td>busking</td>
</tr>
</tbody>
</table>

*from Levy & Goldberg (2014)*
## Qualitative Evaluation

- Hogwards: domain vs semantic type (famous schools)

<table>
<thead>
<tr>
<th>target word</th>
<th>BoW5</th>
<th>BoW2</th>
<th>DEPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>hogwarts</td>
<td>dumbledore</td>
<td>evernight</td>
<td>sunnydale</td>
</tr>
<tr>
<td>hallows</td>
<td>sunnydale</td>
<td>collinwood</td>
<td></td>
</tr>
<tr>
<td>half-blood</td>
<td>garderobe</td>
<td>calarts</td>
<td></td>
</tr>
<tr>
<td>malfoy</td>
<td>blandings</td>
<td>greendale</td>
<td></td>
</tr>
<tr>
<td>snape</td>
<td>collinwood</td>
<td>millfield</td>
<td></td>
</tr>
</tbody>
</table>
Qualitative Evaluation

- Florida: bag-of-words contexts generate meronyms (counties or cities within Florida), while dependency-based contexts provide cohyponyms (other US states)

<table>
<thead>
<tr>
<th>target word</th>
<th>BoW5</th>
<th>BoW2</th>
<th>DePS</th>
</tr>
</thead>
<tbody>
<tr>
<td>florida</td>
<td>gainesville</td>
<td>fla</td>
<td>texas</td>
</tr>
<tr>
<td></td>
<td>fla</td>
<td>alabama</td>
<td>louisiana</td>
</tr>
<tr>
<td></td>
<td>jacksonville</td>
<td>gainesville</td>
<td>georgia</td>
</tr>
<tr>
<td></td>
<td>tampa</td>
<td>tallahassee</td>
<td>california</td>
</tr>
<tr>
<td></td>
<td>lauderdale</td>
<td>texas</td>
<td>carolina</td>
</tr>
</tbody>
</table>
Qualitative Evaluation

- object-oriented, dancing: dep-based embeddings share a syntactic function (adjectives, gerunds)

<table>
<thead>
<tr>
<th>target word</th>
<th>BoW5</th>
<th>BoW2</th>
<th>DEPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>object-oriented</td>
<td>aspect-oriented</td>
<td>aspect-oriented</td>
<td>event-driven</td>
</tr>
<tr>
<td></td>
<td>smalltalk</td>
<td>event-driven</td>
<td>domain-specific</td>
</tr>
<tr>
<td></td>
<td>event-driven</td>
<td>objective-c</td>
<td>rule-based</td>
</tr>
<tr>
<td></td>
<td>prolog</td>
<td>dataflow</td>
<td>data-driven</td>
</tr>
<tr>
<td></td>
<td>domain-specific</td>
<td>4gl</td>
<td>human-centered</td>
</tr>
</tbody>
</table>
Qualitative Evaluation

- object-oriented, dancing: dep-based embeddings share a syntactic function (adjectives, gerunds)

<table>
<thead>
<tr>
<th>target word</th>
<th>BoW5</th>
<th>BoW2</th>
<th>DEPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>dancing</td>
<td>singing</td>
<td>singing</td>
<td>singing</td>
</tr>
<tr>
<td>dance</td>
<td>dance</td>
<td>dance</td>
<td>rapping</td>
</tr>
<tr>
<td>dances</td>
<td>dances</td>
<td>dances</td>
<td>breakdancing</td>
</tr>
<tr>
<td>dancers</td>
<td>breakdancing</td>
<td>miming</td>
<td></td>
</tr>
<tr>
<td>tap-dancing</td>
<td>clowning</td>
<td></td>
<td>busking</td>
</tr>
</tbody>
</table>
Qualitative Evaluation

- object-oriented, dancing: dep-based embeddings share a syntactic function (adjectives, gerunds)

<table>
<thead>
<tr>
<th>target word</th>
<th>BoW5</th>
<th>BoW2</th>
<th>DEPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>dancing</td>
<td>singing</td>
<td>singing</td>
<td>singing</td>
</tr>
<tr>
<td>dance</td>
<td>dance</td>
<td>dance</td>
<td>rapping</td>
</tr>
<tr>
<td>dances</td>
<td>dances</td>
<td>dances</td>
<td>breakdancing</td>
</tr>
<tr>
<td>dancers</td>
<td>breakdancing</td>
<td>miming</td>
<td></td>
</tr>
<tr>
<td>tap-dancing</td>
<td>clowning</td>
<td>busking</td>
<td></td>
</tr>
</tbody>
</table>

Larger window size → more topicality
Quantitative Evaluation: WordSim353

- Word pairs that show
  - relatedness (topical similarity)
  - similarity (functional similarity)

- Task setup
  - rank the *similar* pairs above the *related* ones
  - ranking according to cosine similarity between embeddings
  - draw recall-precision curve that describes the embedding’s affinity towards one subset over another
Dependency-based embeddings

FastText

Quantitative Evaluation: WordSim353

• Word pairs that show
  • relatedness (topical similarity)
  • similarity (functional similarity)

• Task setup
  • rank the similar pairs above the related ones
  • ranking according to cosine similarity between embeddings
  • draw recall-precision curve that describes the embedding’s affinity towards one subset over another

What behaviour would you expect?
Quantitative Evaluation: WordSim353

- Word pairs that show
  - relatedness (topical similarity)
  - similarity (functional similarity)

- Task setup
  - rank the *similar* pairs above the *related* ones
  - ranking according to cosine similarity between embeddings
  - draw recall-precision curve that describes the embedding’s affinity towards one subset over another

Expectation: Curve for **DEPS** > **BoW2** > **BoW5**
• Recall-precision curve when ranking similar words above related words
  (a) based on WordSim353 dataset
  (b) based on Chiarello et al. (1990) dataset (domain vs. function)
What results would you expect when using dependency-based embeddings for the analogy task?

- Recall-precision curve when ranking similar words above related words
  - (a) based on WordSim353 dataset
  - (b) based on Chiarello et al. (1990) dataset (domain vs. function)

From Levy & Goldberg (2014)
Quantitative Evaluation: WordSim353

- Recall-precision curve when ranking similar words above related words
  (a) based on WordSim353 dataset
  (b) based on Chiarello et al. (1990) dataset (domain vs. function)

What results would you expect when using dependency-based embeddings for the analogy task? Dependencies worse than BoW for analogies.
Insights into the model

- Neural word embeddings are often considered uninterpretable, unlike sparse, count-based distributional representations where each dimension corresponds to a particular known context
  ⇒ not possible to assign a meaning to each dimension

How can we get insights into neural word embeddings?
Insights into the model

- Neural word embeddings are often considered uninterpretable, unlike sparse, count-based distributional representations where each dimension corresponds to a particular known context ⇒ not possible to assign a meaning to each dimension

How can we get insights into neural word embeddings?

- Examine which contexts are activated by a target word
- Model learns to maximise the dot product $v_c \cdot v_w$ for observed word pairs $(w, c)$
  - Keep context embeddings
  - Which contexts are most activated by a given target word (i.e.: have the highest dot product)
## Insights into the model

- List 5 most activated contexts for example words
- Most discriminative syntactic contexts

<table>
<thead>
<tr>
<th>batman</th>
<th>hogwarts</th>
<th>turing</th>
</tr>
</thead>
<tbody>
<tr>
<td>superman/conj^-1</td>
<td>students/prep_at^-1</td>
<td>machine/nn^-1</td>
</tr>
<tr>
<td>spider-man/conj^-1</td>
<td>educated/prep_at^-1</td>
<td>test/nn^-1</td>
</tr>
<tr>
<td>superman/conj</td>
<td>student/prep_at^-1</td>
<td>theorem/poss^-1</td>
</tr>
<tr>
<td>spider-man/conj</td>
<td>stay/prep_at^-1</td>
<td>machines/nn^-1</td>
</tr>
<tr>
<td>robin/conj</td>
<td>learned/prep_at^-1</td>
<td>tests/nn^-1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>florida</th>
<th>object-oriented</th>
<th>dancing</th>
</tr>
</thead>
<tbody>
<tr>
<td>marlins/nn^-1</td>
<td>programming/amod^-1</td>
<td>dancing/conj</td>
</tr>
<tr>
<td>beach/appos^-1</td>
<td>language/amod^-1</td>
<td>dancing/conj^-1</td>
</tr>
<tr>
<td>jacksonville/appos^-1</td>
<td>framework/amod^-1</td>
<td>singing/conj^-1</td>
</tr>
<tr>
<td>tampa/appos^-1</td>
<td>interface/amod^-1</td>
<td>singing/conj</td>
</tr>
<tr>
<td>florida/conj^-1</td>
<td>software/amod^-1</td>
<td>ballroom/nn</td>
</tr>
</tbody>
</table>

From Levy & Goldberg (2014)
Generalisation of SGNS

Sum-up

• Generalisation of linear bag-of-words context to arbitrary contexts
  • here: dependency-based contexts
• Depending on the context, the model learns different properties from the same data
Generalisation of SGNS

Sum-up

- Generalisation of linear bag-of-words context to arbitrary contexts
  - here: dependency-based contexts
- Depending on the context, the model learns different properties from the same data

Dependency-based embeddings

- are less topical and exhibit more functional similarity than the original skipgram embeddings
Generalisation of SGNS

Sum-up

- Generalisation of linear bag-of-words context to arbitrary contexts
  - here: dependency-based contexts
- Depending on the context, the model learns different properties from the same data

Dependency-based embeddings

- are less topical and exhibit more functional similarity than the original skipgram embeddings

What other contexts are possible?
FastText – Background

- Mikolov et al. 2013: Distributed Representations of words and phrases and their compositionality
  - Representation of words in vector space
  - Drawbacks:
    - no sentence representations
    - does not exploit morphology
      (different representations for disaster / disastrous)
FastText – Motivation

- Better representations for morphological variants of same word
- Better representations for rare/unseen words

⇒ Train word representations with character-level features
FastText – Motivation

- Better representations for morphological variants of same word
- Better representations for rare/unseen words

⇒ Train word representations with character-level features

Ich habe einen Apfel **gegessen**

- Use character ngrams to predict surrounding context
Recap: SkipGram

### Model probability of a context word given a word representation

- For word $w$: $v_w$
- For context word $c$: $v_c$

$$p(c | w) = e^{v_w^\top v_c} \sum_{k=1}^{K} e^{v_w^\top v_k}$$

### Word vectors

- $v_w \in \mathbb{R}^d$

### Softmax

- Computationally expensive
- Use approximations:
  - Hierarchical softmax
  - Negative sampling

$$\log(1 + e^{-v_w^\top w_t}) + \sum_{n \in N_c} \log(1 + e^{-v_w^\top w_n})$$
Recap: SkipGram

- Model probability of a context word given a word

representation for word \( w \): \( v_w \)
representation for context word \( c \): \( v_c \)

- Word vectors \( v_w \in \mathbb{R}^d \)

\[
p(c|w) = \frac{e^{v_w^T v_c}}{\sum_{k=1}^{K} e^{v_w^T v_k}}
\]
Recap: SkipGram

- Model probability of a context word given a word

representation for word $w$: $v_w$

representation for context word $c$: $v_c$

- Word vectors $v_w \in \mathbb{R}^d$

- Softmax computationally expensive

$\Rightarrow$ use approximations:

- Hierarchical softmax

- Negative sampling $\log(1 + e^{-v_w^T v_c}) + \sum_{n \in N_c} \log(1 + e^{v_w^T v_n})$
Recap: CBOW

$P(w | C) = \frac{e^{h_C^T v_w}}{\sum_{k=1}^K e^{h_C^T v_k}}$
Recap: CBOB

Model probability of a word given the context

representation for context $C$: $h_c$

representation for word $w$: $v_w$

Continuous bag of words

(sum of the words in the context)
FastText

- As in SkipGram: model probability of a context word $c$ given a word $w$
  
  representation for word $w$: $h_w$
  
  representation for word $c$: $v_c$

  $$p(c|w) = \frac{e^{h_w^T v_c}}{\sum_{k=1}^{K} e^{h_w^T v_k}}$$
FastText

- As in SkipGram: model probability of a context word $c$ given a word $w$

  
  representation for word $w$: $h_w$
  
  representation for word $c$: $v_c$

  
  
  
  \[
  p(c|w) = \frac{e^{h_w^T v_c}}{\sum_{k=1}^{K} e^{h_w^T v_k}}
  \]

- Representation of a word $w$ computed based on ngrams: all ngrams with length $l$ where $3 \leq l \leq 6$ and word form
FastText

- As in SkipGram: model probability of a context word $c$ given a word $w$

representation for word $w$: $h_w$
representation for word $c$: $v_c$

$$p(c|w) = \frac{e^{h_w^T v_c}}{\sum_{k=1}^{K} e^{h_w^T v_k}}$$

- Representation of a word $w$ computed based on ngrams: all ngrams with length $l$ where $3 \leq l \leq 6$ and word form

$$h_w = \sum_{g_l \in w} v_g$$

$3 \leq l \leq 6$
Advantages of FastText

Out-of-Vocabulary (OOV) words

- Ngram representations are shared across words
  ⇒ more reliable representations for rare words
- We now can build vectors for unseen words:
Advantages of FastText

Out-of-Vocabulary (OOV) words

- Ngram representations are shared across words ⇒ more reliable representations for rare words
- We now can build vectors for unseen words:

\[ h_w = \sum_{g \in w} v_g \]

\[
\begin{align*}
&\text{ver} \\
&\text{ker} \quad \text{wink} \\
&\text{zwink} \quad \text{ink} \\
&\text{kert} \quad \text{inke} \\
&\text{ert} \quad \ldots \\
\end{align*}
\]

char ngrams \hspace{1cm} \text{word form}
FastText Training

- Training with Stochastic Gradient Descent
- Minimise negative log-likelihood
- Set ngram length $= 0$
FastText Training

- Training with Stochastic Gradient Descent
- Minimise negative log-likelihood
- Set ngram length $= 0 \Rightarrow$ SkipGram with negative sampling
FastText Training

• Training with Stochastic Gradient Descent
• Minimise negative log-likelihood
• Set ngram length = 0 ⇒ SkipGram with negative sampling

• Evaluation – model parameters:
  • 300 dimensions
  • sample 5 negative examples per word
  • context window size $c$, uniformly sample $c$ between 1 and 5
  • subsample frequent words with threshold $10^{-4}$
  • discard all words that occur $<$ 5 times in the corpus
  • learning rate 0.05

Training speed:
Model is around 1.5 × slower than SkipGram
FastText Training

- Training with Stochastic Gradient Descent
- Minimise negative log-likelihood
- Set ngram length = 0 $\Rightarrow$ SkipGram with negative sampling

Evaluation – model parameters:
- 300 dimensions
- sample 5 negative examples per word
- context window size $c$, uniformly sample $c$ between 1 and 5
- subsample frequent words with threshold $10^{-4}$
- discard all words that occur $< 5$ times in the corpus
- learning rate 0.05

Training speed:
- Model is around $1.5 \times$ slower than SkipGram
Word Similarity Evaluation

- Given: pair of words $w_1, w_2$
- Compare cosine similarity for $w_1, w_2$ against human judgements
  
  \[ s(w_1, w_2) = \frac{x_{w1}^T x_{wq}}{||x_{w1}|| \cdot ||x_{w2}||} \]

- Spearman's rank correlation

<table>
<thead>
<tr>
<th>Language</th>
<th>Method</th>
<th>SG</th>
<th>CBOW</th>
<th>FT*</th>
<th>FT</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR</td>
<td>WS353</td>
<td>51</td>
<td>52</td>
<td>54</td>
<td>55</td>
</tr>
<tr>
<td>DE</td>
<td>GUR350</td>
<td>61</td>
<td>62</td>
<td>64</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>GUR65</td>
<td>78</td>
<td>78</td>
<td>81</td>
<td>81</td>
</tr>
<tr>
<td></td>
<td>ZG222</td>
<td>35</td>
<td>38</td>
<td>41</td>
<td>44</td>
</tr>
<tr>
<td>EN</td>
<td>RW</td>
<td>43</td>
<td>43</td>
<td>46</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>WS353</td>
<td>72</td>
<td>73</td>
<td>71</td>
<td>71</td>
</tr>
<tr>
<td>ES</td>
<td>WS353</td>
<td>57</td>
<td>58</td>
<td>58</td>
<td>59</td>
</tr>
<tr>
<td>FR</td>
<td>RG65</td>
<td>70</td>
<td>69</td>
<td>75</td>
<td>75</td>
</tr>
<tr>
<td>RO</td>
<td>WS353</td>
<td>48</td>
<td>52</td>
<td>51</td>
<td>54</td>
</tr>
<tr>
<td>RU</td>
<td>HJ</td>
<td>59</td>
<td>60</td>
<td>60</td>
<td>66</td>
</tr>
</tbody>
</table>

FT* uses null vector for unknowns.
Word Similarity Evaluation

- Given: pair of words \( w_1, w_2 \)
- Compare cosine similarity for \( w_1, w_2 \) against human judgements
  \[
s(w_1, w_2) = \frac{x_{w_1}^T x_{w_q}}{||x_{w_1}|| \cdot ||x_{w_2}||}
\]
- Spearman’s rank correlation

<table>
<thead>
<tr>
<th>Language</th>
<th>Method</th>
<th>SG</th>
<th>CBOW</th>
<th>FT*</th>
<th>FT</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR</td>
<td>WS353</td>
<td>51</td>
<td>52</td>
<td>54</td>
<td>55</td>
</tr>
<tr>
<td>DE</td>
<td>GUR350</td>
<td>61</td>
<td>62</td>
<td>64</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>GUR65</td>
<td>78</td>
<td>78</td>
<td>81</td>
<td>81</td>
</tr>
<tr>
<td></td>
<td>ZG222</td>
<td>35</td>
<td>38</td>
<td>41</td>
<td>44</td>
</tr>
<tr>
<td>EN</td>
<td>RW</td>
<td>43</td>
<td>43</td>
<td>46</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>WS353</td>
<td>72</td>
<td>73</td>
<td>71</td>
<td>71</td>
</tr>
<tr>
<td>ES</td>
<td>WS353</td>
<td>57</td>
<td>58</td>
<td>58</td>
<td>59</td>
</tr>
<tr>
<td>FR</td>
<td>RG65</td>
<td>70</td>
<td>69</td>
<td>75</td>
<td>75</td>
</tr>
<tr>
<td>RO</td>
<td>WS353</td>
<td>48</td>
<td>52</td>
<td>51</td>
<td>54</td>
</tr>
<tr>
<td>RU</td>
<td>HJ</td>
<td>59</td>
<td>60</td>
<td>60</td>
<td>66</td>
</tr>
</tbody>
</table>

FT* uses null vector for unknowns

Works particularly well for datasets with rare words and for morphologically rich languages
Word Analogy Evaluation

- Paris $\rightarrow$ France; Rom $\rightarrow$ ?
  - Predict the analogy
  - Evaluate using accuracy

What results would you expect?
Word Analogy Evaluation

- Paris → France; Rom → ?
  - Predict the analogy
  - Evaluate using accuracy

<table>
<thead>
<tr>
<th>Language</th>
<th>Semantic</th>
<th>Syntactic</th>
<th>SG</th>
<th>CBOW</th>
<th>FT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>25.7</td>
<td>52.8</td>
<td>27.6</td>
<td>55.0</td>
<td>77.8</td>
</tr>
<tr>
<td>DE</td>
<td>66.5</td>
<td>44.5</td>
<td>66.8</td>
<td>45.0</td>
<td>56.4</td>
</tr>
<tr>
<td>EN</td>
<td>78.5</td>
<td>70.1</td>
<td>78.2</td>
<td>69.9</td>
<td>74.9</td>
</tr>
<tr>
<td>IT</td>
<td>52.3</td>
<td>51.5</td>
<td>54.7</td>
<td>51.8</td>
<td>62.7</td>
</tr>
</tbody>
</table>
Word Analogy Evaluation

- Paris → France; Rom → ?
  - Predict the analogy
  - Evaluate using accuracy

<table>
<thead>
<tr>
<th></th>
<th>SG</th>
<th>CBOW</th>
<th>FT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>Semantic 25.7</td>
<td>27.6</td>
<td>27.5</td>
</tr>
<tr>
<td></td>
<td>Syntactic 52.8</td>
<td>55.0</td>
<td>77.8</td>
</tr>
<tr>
<td>DE</td>
<td>Semantic 66.5</td>
<td>66.8</td>
<td>62.3</td>
</tr>
<tr>
<td></td>
<td>Syntactic 44.5</td>
<td>45.0</td>
<td>56.4</td>
</tr>
<tr>
<td>EN</td>
<td>Semantic 78.5</td>
<td>78.2</td>
<td>77.8</td>
</tr>
<tr>
<td></td>
<td>Syntactic 70.1</td>
<td>69.9</td>
<td>74.9</td>
</tr>
<tr>
<td>IT</td>
<td>Semantic 52.3</td>
<td>54.7</td>
<td>52.3</td>
</tr>
<tr>
<td></td>
<td>Syntactic 51.5</td>
<td>51.8</td>
<td>62.7</td>
</tr>
</tbody>
</table>

- Works well for syntactic analogies, especially for morphologically rich languages (CS, DE)

  groß → größer; hoch → ?
Effect of training data size

- FastText works well for rare and unknown words
- **Hypothesis**: FastText is also better in settings where we do not have a lot of training data.
- **Test**: train CBOW and FastText on subsets of Wikipedia (1, 2, 5, 10, 20, 50%)
Effect of training data size

- FastText works well for rare and unknown words
- **Hypothesis**: FastText is also better in settings where we do not have a lot of training data.
- **Test**: train CBOW and FastText on subsets of Wikipedia (1, 2, 5, 10, 20, 50%)

![Graphs showing Spearman rank comparison between CBOW and FastText](image)
Effect of training data size

- FastText works well for rare and unknown words
- **Hypothesis**: FastText is also better in settings where we do not have a lot of training data.
- **Test**: train CBOW and FastText on subsets of Wikipedia (1, 2, 5, 10, 20, 50%)

Adding more data does not always improve results
Word similarity evaluation for unknown words

- Train on 1% of EN Wikipedia
- Report cosine similarity for n-grams of word pairs where one word is unknown
Word similarity evaluation for unknown words

- Train on 1% of EN Wikipedia
- Report cosine similarity for ngrams of word pairs where one word is unknown
FastText – Sum-up

• Extension of the SGNS model that represents each word by the sum of its subword representations
• For ngram length=0 ⇒ same as SGNS
• Fast to train, good results for smaller training data sizes
• Superior performance especially for rare and unknown words and for syntactic analogies
References

- Omer Levy and Yoav Goldberg (2014): Dependency-based word embeddings. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 302–308, Baltimore, Maryland, USA