PythonとKerasによるディープラーニング

第6章 テキストとシーケンスのためのディープラーニング
6.1 テキストデータの操作

19ss317u　　ZHAO　YI

6.1 テキストデータの操作

- テキストを数値テンソルに変換するプロセスはテキストのベクトル化である。
- テキストのベクトル化は複数の方法で行うことができる。

トークン化

a.テキストを**単語**に分割し、各単語をベクトルに変換する。

b.テキストを**文字**に分割し、各文字をベクトルに変換する。

c.**Nグラム**の単語または文字を抽出し、各Nグラムをベクトルに変換する。

ベクトルにトークンを関連付ける方法

- 主な方法：

    1)　one-hotエンコーディング

    2)トークン埋め込み（単語埋め込み）

6.1.1単語と文字のone-hotエンコーディング

- One-hotエンコーディング：各単語に一意な整数のインデックスを割り当て、この整数のインデックスiをサイズNの二値ベクトルに変換する。また、このベクトルはi番目のエントリが1である以外はすべて 0 に設定される。

  eg. 1)私が犬です。

  2)僕は犬が好きです。

  [私，犬，僕，好き]
  - 1)[1,1,0,0]
  - 2)[0,1,1,1]

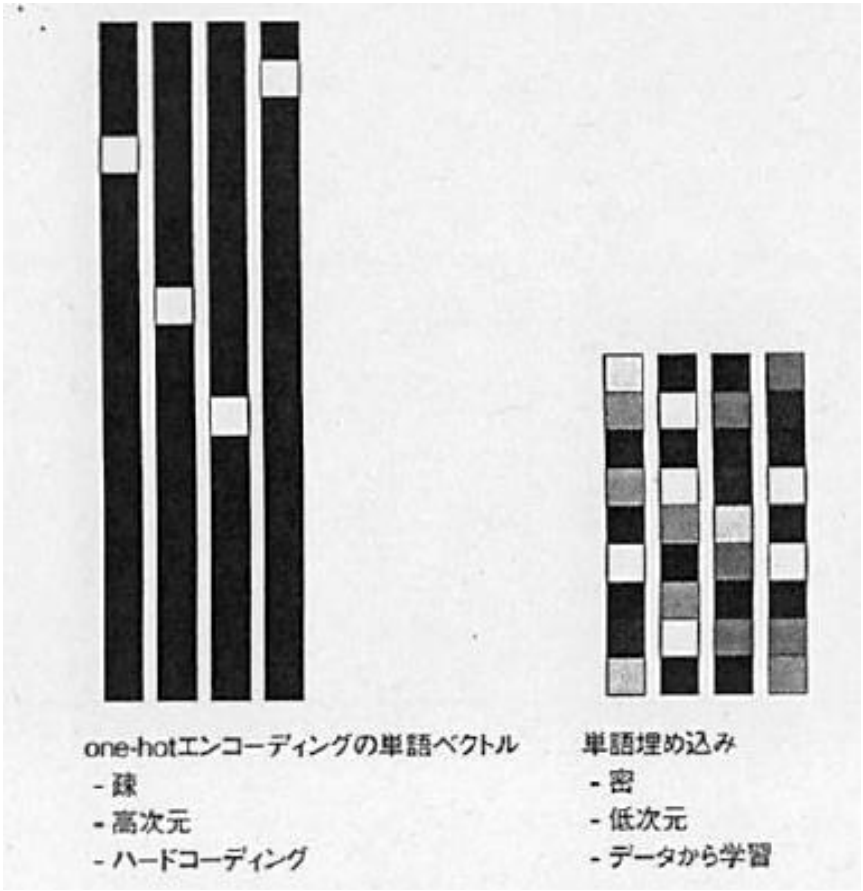
- Kerasには、生のテキストデータに単語または文字レベルでone-hotエンコーディングを適用するためのユーティリティが組み込まれているので、簡単に実行できる。

One-hotハッシュトリック

- One-hotエンコーディングの一種である。
- 意味：各単語にインデックスを明示的に割り当て、インデックスをディクショナリで参照する代わりに、単語を固定サイズのベクトルにハッシュ化できる。
- 利点：メモリを節約し、データのオンラインエンコーディングを可能にすることです。
- 欠点：ハッシュ衝突---2つの異なる単語のハッシュが同じになってしまって、これらのハッシュを調べる機械学習モデルがそれらの単語の違いを区別できなくなる可能性がある。

6.1.2 単語埋め込み

- 単語をベクトルに関連付けるもう1つの強力な手法は密な単語ベクトルの使用である。
- One-hotエンコーディングやハッシュトリックによって得られる単語表現　と　単語埋め込みによって得られる単語表現を比べる図
- つまり、単語埋め込みのほうが少ない次元数でより多くの情報を格納できる。



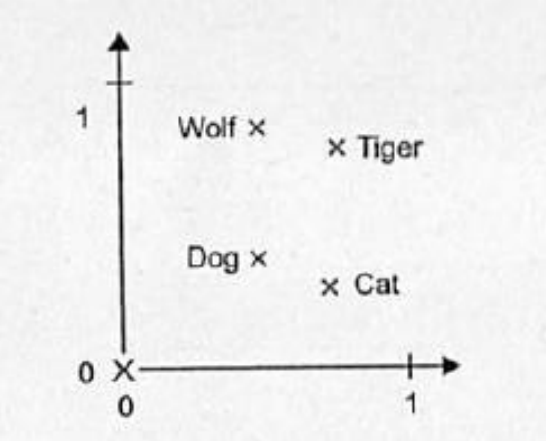
単語埋め込みを取得する方法

1)メインのタスクと同時に単語埋め込みを学習する。

2)別の機械学習タスクを使って計算された単語埋め込みをモデルに読み込む。(学習済みの単語埋め込み)

1)埋め込み層を使った単語埋め込みの学習

- 問題点：得られる埋め込み空間が構造的ではないことである。理解するのは困難である。
- 単語埋め込みは、人間の言語を幾何学的にマッピングするためのものである。距離に加えて、埋め込み空間の向きに意味を持たせる。



cat → tiger
dog → wolf
pet → wild
animal

dog → cat
wolf → tiger
イヌ科 → ネコ科

埋め込み層(Embedding層)

単語のインデックス⟶埋め込み層⟶対応する単語ベクトル

Embedding層：整数のインデックスを密ベクトルにマッピングするディクショナリとして考えてみるのが一番である。

- インデックス化する際には、層の重みもランダムに初期化される。
- 訓練の際には、単語ベクトルがバックプロパゲーションを通じて少しずつ調整され、下流のモデルで利用可能な空間が形作られていく。
- 訓練が完了した時点で埋め込み空間は構造化された空間になる。