1 Введение

В лексической системе русского языка есть слова, которые звучат одинаково, но имеют совершенно разные значения. Такие слова называются лексическими омонимами, а звуковое и грамматическое совпадение разных языковых единиц, которые семантически не связаны друг с другом, называется омонимией.

Например:

1. Ключ(1) – «Родник» (Студеный ключ)

   Ключ(2) – Стальной ключ для отпирания и запирания замка

2. Лук(1) – «Растение» зеленый лук

   Лук(2) – «оружие» тугой лук

В отличие от многозначных слов лексические омонимы не обладают предметно-семантической связью, т.е. у них нет общих семантических признаков, по которым можно было бы судить о полисемантизме одного слова. Полная лексическая омонимия – это совпадение слов, принадлежащих к одной части речи, во всех формах. Пример: наряд(1) – «одежда», наряд(2) – «распоряжение»; Замок(1) – «123», Замок(2) – «здание»;

Автоматическая система перевода делится на несколько этапов, одним из которых является морфологический. На этом этапе для каждого слова определяются морфологические характеристики: род, число, падеж, склонение, и т.п., а также начальная форма слова (Лемма). Процесс морфологической разметки осложняется омонимами.

2 Методы представления текста

В настоящее время теория и практика машинного обучения переживают настоящую «глубинную революцию», вызванную успешным применением методов deep learning, представляющих собой третье поколение нейронных сетей. Сети, обученные с помощью алгоритмов глубинного обучения, не просто превзошли по точности лучшие альтернативные подходы, но и в ряде задач проявили зачатки понимания смысла подаваемой информации (например, при распознавании изображений, анализе текстовой информации и т.п.). Наиболее успешные современные промышленные методы компьютерного зрения, распознавания речи и машинного перевода построены на использовании глубинных сетей, а гиганты IT-индустрии, такие как Apple, Google, Facebook, скупают коллективы исследователей, занимающихся глубинным обучением целыми лабораториями.

Представление слова как вектор - в настоящее время одна из самых интересных областей исследований в глубинном обучении, хотя данный подход был изначально введен Bengio[1] и др. более десяти лет назад.

Самый известные модели представления слова как вектор представлены Mikolov (2013a) и реализация моделей Word2vec[2] опубликована на сайте Google project, которая привлекла большое внимание в последние два года – это “Continuous-Bag-of-Word” и “Skip-Gram”. Эти модели использовали простую нейронную сеть для отображения слов, которые ближе к слову по значению. Поэтому процесс обучения для этих моделей быстро работает с большими данными.

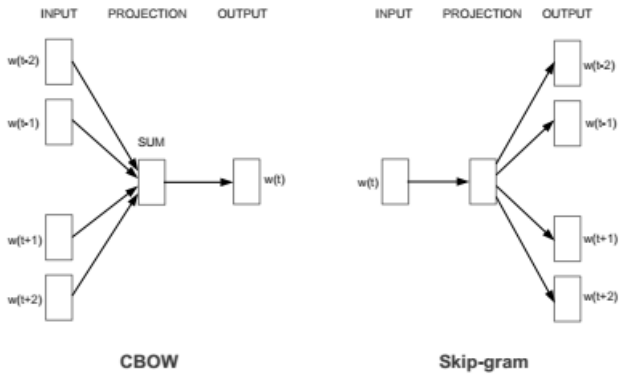

Рис. 1: Модели CBOW и Skip-gram.

CBOW модель – Использует окружающие слова (предложение без заданного слова), что бы предугадать заданное слово. Входные вектора в нейронную сеть W(t-2),W(t-1), W(t+1), W(t+2), а выходной вектор из сети это заданное слово W(t). В результате обучения получается 2 вектора представления слова. Входной вектор V - это вектор представления предложения описания слова, выходной вектор V’ – Это вектор представления слова.

SKIP-GRAM модель – Использует слово, что бы предугадать предложение описания заданного слова.

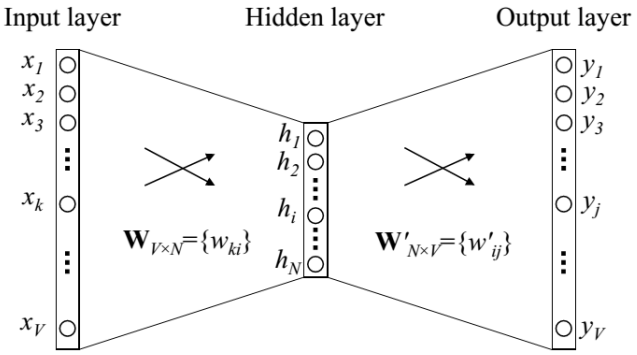

Рис. 2: Показал нейронную сеть, использованную чтобы получить вектор представления слова. Входной слой и выходной имеют V нейронов, а скрыты имеет N нейронов.

Эти модели могут использоваться во многих задачах анализа естественного языка, таких как классификация документов, распознавание спама или машинный перевод.

На рисунке 3, визуализировали векторы для чисел и животных на английском и испанском языках, и он может быть легко видеть, что эти понятия имеют схожие геометрические композиции. Причина в том, что, как и все распространенные языки поделиться понятия, которые основаны на реальном мире

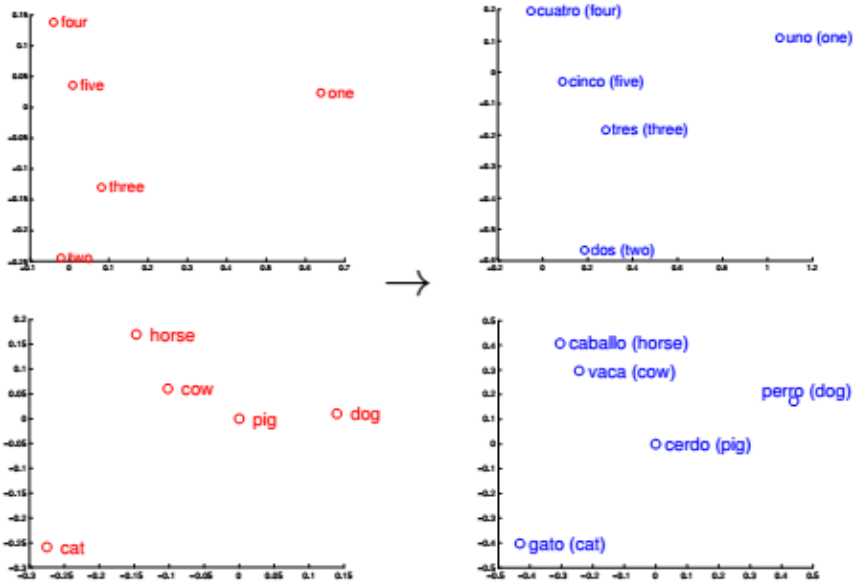

Рис. 3: Распределенная векторных представлений слов чисел и животных на английском языке (слева) и испанском (справа).пять Векторов на каждом языке были спроектированы до двух измерениях с использованием PCA. Видно, что эти концепции имеют схожие механизмы геометрические в обоих пространствах, предполагая, что можно узнать точную линейное отображение из одного пространства в другое.

Один хороший пример это двуязычный слово вложение, производится в Socher др. (2013a). Мы можем научиться вставлять слова из двух разных языков в одной общей пространстве. В этом случае, мы учимся вставлять английские и китайский слов в том же пространстве.

Одна вещь, мы можем сделать, чтобы почувствовать слова вложения пространства для визуализации их трет-СНЭ, сложной техники для визуализации многомерных данных

Таким образом, возможно снять омонимию через окружающие слова в предложении. Если значение слов окружающих омонимы разные, тогда расстояние

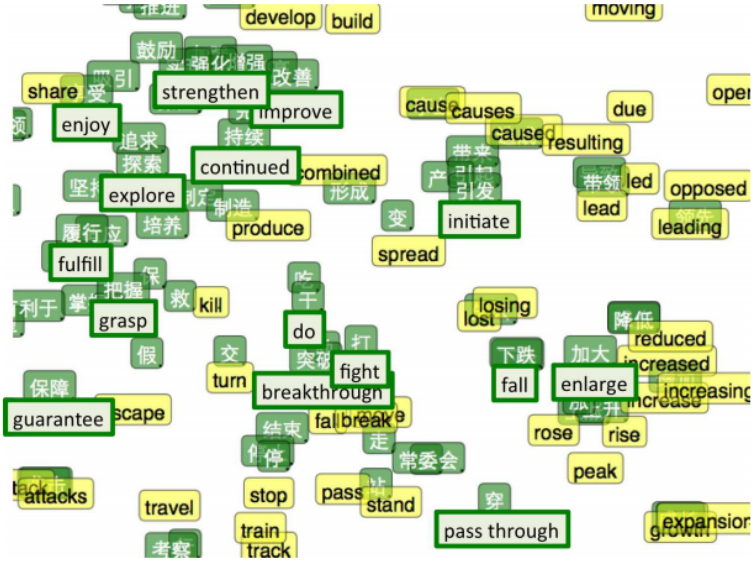

Рис. 4: Английский и китайский язык.

между омонимами тоже большое.

3 Нейронные сети и Автоэнкодер

Нейронные сети прямого распространения часто используются для обучения с учителем и используются, например, для классификации.

Часто используемая функция активации - сигмоид:

$$f(z_j) = \sigma(z_j) = (1 + e^{-z_j})^{-1} \tag{1}$$

Где:

$$z_j = \sum w_{ij} x_i \tag{2}$$

Полезное свойство сигмоида - её производная функции:

$$\frac{\partial \sigma}{\partial z} = \sigma(z)(1 - \sigma(z)) \tag{3}$$

Обучение такой нейронной сети производится обычно методом обратного распространения ошибки таким образом, чтобы минимизировать среднеквадратическую ошибку сети на обучающей выборке. Таким образом, обучающая выборка содержит пары векторов признаков (входные данные) и эталонных векторов (маркированные данные) (x, y).

Метод обратного распространения ошибки:

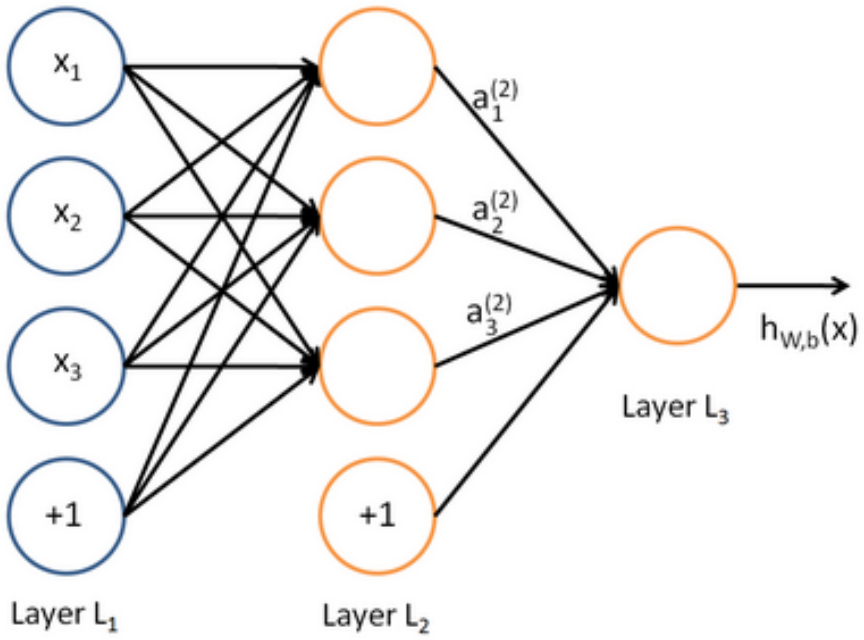

Рис. 5: Многослойная нейронная сеть

Для выходного нейрона:

$$\delta = z - y \tag{4}$$

Для нейронов скрытых слоев:

$$\delta_i = \sum_{j=0}^{n} \delta_j w_{ij} \tag{5}$$

Коррекция весов:
Для выходного нейрона:

$$w'_{i0} = w_{i0} + \eta \delta y_i \tag{6}$$

Для нейронов скрытых слоев:

$$w'_{ij} = w_{ij} + \eta \delta_j y_j (1 - y_j) y_i \tag{7}$$

В реальной практике, маркированных данных очень мало, для них требуется много сил и времени. Автоэнкодер представляет собой алгоритм обучения без учителя, который использует нейронную сеть и метод обратного распространения ошибки для того, чтобы добиться того, что входной вектор признаков вызывал выход сети, равный входному вектору, т.е. y = x.

Автоэнкодер является специальной архитектурой искусственных нейронных сетей, позволяющая применять обучение без учителя при использовании метода

обратного распространения ошибки. Простейшая архитектура автоэнкодера — сеть прямого распространения, без обратных связей, наиболее схожая с перцептроном и содержащая входной слой, скрытый слой и выходной слой. В отличие от перцептрона, выходной слой автоэнкодера должен содержать столько же нейронов, сколько и входной слой.

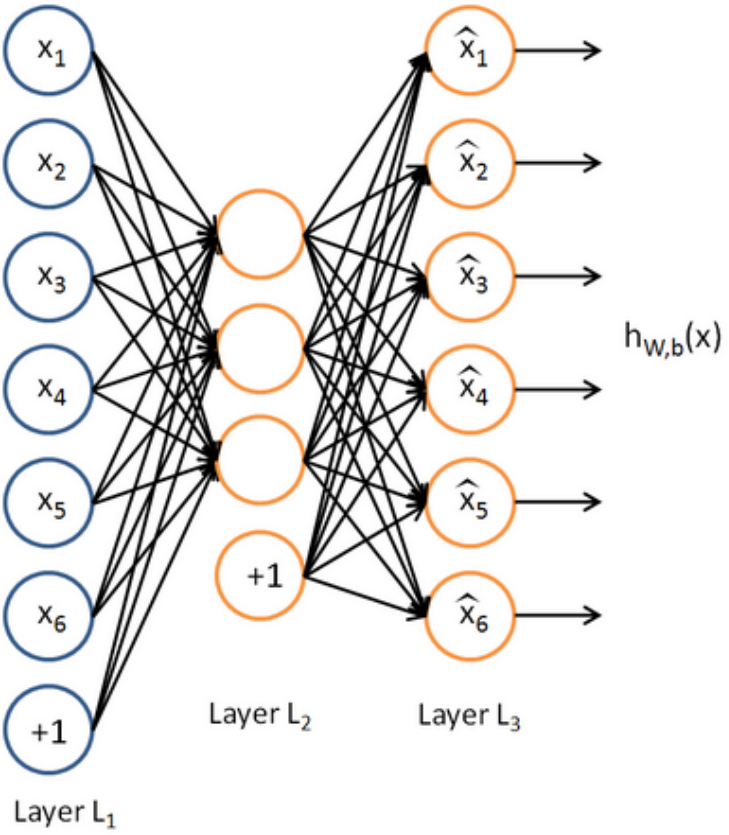

Рис. 6: Автоэнкодер

Цель автоэнкодер - чтобы выход нейронной сети был наиболее близким к входному вектору. Для того, чтобы решение этой задачи было нетривиальным, на топологию сети накладываются особые условия:

1. Количество нейронов скрытого слоя должно быть меньше, чем размерность входных данных.

2. Активация нейронов скрытого слоя должна быть разреженной.

Первое ограничение позволяет получить сжатие данных при передаче входного сигнала на выход сети. Такая сжатие возможно, если в данных есть скрытые взаимосвязи, корреляция признаков или структура.

Второе ограничение – требование разреженной активации нейронов скрытого слоя, — позволяет получить нетривиальные результаты даже когда количество нейронов скрытого слоя превышает размерность входных данных. Будем считать нейрон активным, когда значение его функции передачи близко к 1 и наоборот,

неактивным если значение его функции передачи близко к 0. Разреженная активация – это когда количество неактивных нейронов в скрытом слое значительно превышает количество активных.

Эти ограничения заставляют нейросеть искать обобщения и корреляцию в поступающих на вход данных, выполнять их сжатие. Таким образом, нейросеть автоматически обучается выделять из входных данных общие признаки, которые кодирутся в значениях весов сети.

Мы хотим, чтобы средняя активация каждого скрытого нейронна приняла значение, наиболее близкое к заданному разреженному параметру (порядка 0.05). Для этого, мы добавим в каждый нейрон скрытого слоя параметр разреженности ρ:

$$\hat{\rho}_j = \frac{1}{m} \sum_{i=1}^{m} \left[a_j^{(2)}(x^{(i)}) \right] \tag{8}$$

Мы хотим, чтобы средняя активация каждого скрытого нейронна приняла значение, наиболее близкое к ρ:

$$\hat{\rho}_j = \rho \tag{9}$$

Штравная функция:

$$S = \sum_{j=1}^{s_2} KL(\rho|\hat{\rho}_j) \tag{10}$$

$$KL(\rho|\hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_j} \tag{11}$$

Производная штравной функции:

$$\frac{\partial KL(\rho|\hat{\rho}_j)}{\partial \rho_j} = -\frac{\rho}{\hat{\rho}_j} + \frac{1-\rho}{1-\hat{\rho}_j} \tag{12}$$

4 Кластеризация значений слова

4.1 K-means

K-means является одним из простейших алгоритмов обучения без учителя, который хорошо решает задачу кластеризации. Основная идея состоит в определении K центров, по одному для каждого кластера. Эти центры должны быть помещены в хитро из-за другого места вызывает различный результат. Таким образом, лучшим выбором является размещение их как можно больше далеко друг от друга.

Следующим шагом является принятие каждую точку, принадлежащую к данному набору данных и связать его в ближайший центр. В этот момент мы должны пересчитать K новые центроиды, как барицентра кластеров в результате

предыдущего шага. После того как мы эти новые K центроиды, новую привязку должно быть сделано между теми же набора данных точек и ближайшей нового центра. Цикл был сформирован. В результате этого цикла можно заметить, что K-центры меняют шаг за шагом местоположения пока больше изменений не сделано или, другими словами центры не двигаться больше. Наконец, этот алгоритм направлен на минимизацию целевой функции, как квадрат знаю функцию ошибки, данную функцию:

$$J(V) = \sum_{j}^{c} \sum_{j}^{c_i} (||x_i - v_i||)^2 \tag{13}$$

4.2 Маркировка значений слова

В первых получить все возможные векторы слов, затем используемы их, чтобы построить векторы контекстов с применением автоэнкодер.

2 слова в 1 слово

Пример у нас есть миллион векторов описания о слове Замок, Выпольняем вычисления K-Means для этих веторов. В результате, мы можем переписать кождое слово с его значением, так как замок1, замок2.

Таким образом, все омонимии уже маркированы по значению, мы можем использавать эти резутаты длы задачи снятия омонимий.

5 Список Литературы

[1] [Bengio et al.2003] Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. In Journal of Machine Learning Research, pages 1137–1155.

[2] Национальный корпус русского языка, www.ruscorpora.ru

[3] [Socher et al.2013] Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing with compositional vector grammars. In ACL.

[4] Exploiting Similarities among Languages for Machine Translation.

[5]