Distributed representation of text

He He

New York University

September 22, 2020

Logistics

Fix errors in HW1:

- Question 2.2

$$\frac{d}{d\alpha} \log \sigma(\alpha)$$

- Question 2.5

$$p_N(w) \propto p_{\text{unigram}}^{\beta}(w)$$

Table of Contents

Last week

Generative vs discriminative models for text classification

- (Multinomial) naive Bayes
  - Assumes conditional independence
  - Very efficient in practice (closed-form solution)
- Logistic regression
  - Works with all kinds of features
  - Wins with more data

Features for text

- BoW representation
- N-gram features (usually $n \leq 3$)

Control the complexity of the hypothesis class

- Feature selection
- Norm regularization
- Hyperparameter tuning on the validation set

Evaluation

- Accuracy

$$acc = \frac{\#\ correct}{\#\ total\ pred}$$

	+1	-1
total	10	90
correct	0	90

$$acc = \frac{90}{100} = 0.9$$

- Precision

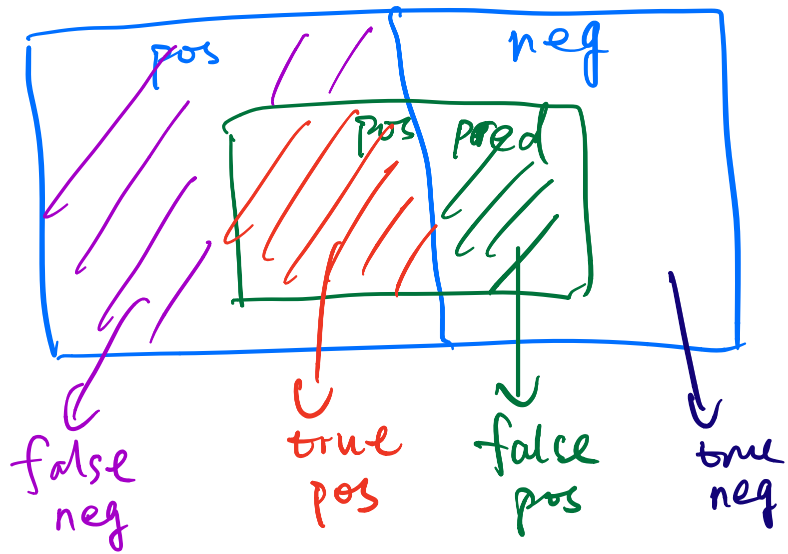
$$\frac{TP}{TP + FP}$$

- Recall

$$\frac{TP}{FN + TP}$$

- F1

$$\frac{2\ precision \times recall}{prec + recall}$$



- Macro vs micro average

Table of Contents

Objective

Goal: come up a good representation of text

- What is a representation?

  - ϕ: text $\rightarrow \mathbb{R}^d$
  - (learned) feature

- What is a good representation?

  - improve task performance
  - proxy: $d(\phi(a), \phi(b))$ is small for semantically similar text a and b.

Distance functions

Let's check if BoW is a good representation.

Euclidean distance

For $a, b \in \mathbb{R}^d$,

$$d(a, b) = \sqrt{\sum_{i=1}^{d} (a_i - b_i)^2} \ .$$

What if b repeats each sentence in a twice?

Distance functions

Let's check if BoW is a good representation.

Euclidean distance

For $a, b \in \mathbb{R}^d$,

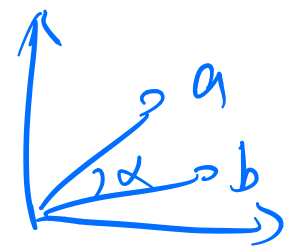
$$d(a, b) = \sqrt{\sum_{i=1}^{d} (a_i - b_i)^2}\ .$$

What if b repeats each sentence in a twice?

Cosine similarity

For $a, b \in \mathbb{R}^d$,

$$\text{sim}(a, b) = \frac{a \cdot b}{\|a\| \|b\|} = \cos \alpha$$



Angle between two vectors

Example: information retrieval

Given a set of documents and a query, use the BoW representation and cosine similarity to find the most relevant document.

What are potential problems?

Example: information retrieval

Given a set of documents and a query, use the BoW representation and cosine similarity to find the most relevant document.

What are potential problems?

- similarity is dominated by common words
- doesn't consider meaning (e.g. synonyms)

Example:

Q: Who has watched Tenet?

She has just watched Joker.

Tenet was shown here last week.

TFIDF

Key idea: upweight words that carry more information about the document

Representation ϕ: document $\to \mathbb{R}^{|\mathcal{V}|}$

TFIDF:

BoW

$$\phi_i(d) = \underbrace{\text{count}(w_i, d)}_{\text{term frequency}} \times \underbrace{\log \frac{\#\text{ documents}}{\#\text{ documents containing } w_i}}_{\text{inverse document frequency}} .$$

TF IDF

Justification:

$$\text{idf}(w, d) = \text{PMI}(w; d) = \log \frac{p(d \mid w)}{p(d)} .$$

$\frac{1}{\#\text{doc cont. } w}$

$\frac{1}{\#\text{docs}}$

Table of Contents

The distributional hypothesis

"You shall know a word by the company it keeps." (Firth, 1957)

Word guessing!

Everybody likes tezgüino.

Takeaway: the meaning of a word can be representated by its neighbors.

The distributional hypothesis

"You shall know a word by the company it keeps." (Firth, 1957)

Word guessing!

Everybody likes tezgüino.

We make tezgüino out of corn.

Takeaway: the meaning of a word can be representated by its neighbors.

The distributional hypothesis

"You shall know a word by the company it keeps." (Firth, 1957)

Word guessing!

Everybody likes tezgüino.

We make tezgüino out of corn.

A bottle of tezgüino is on the table.

Takeaway: the meaning of a word can be representated by its neighbors.

The distributional hypothesis

"You shall know a word by the company it keeps." (Firth, 1957)

Word guessing!

Everybody likes tezgüino.

We make tezgüino out of corn.

A bottle of tezgüino is on the table.

Don't have tezgüino before you drive.

Takeaway: the meaning of a word can be represented by its neighbors.

Choose the context

Where does the neighbors come from? (What relations are we interested in?)

Construct a matrix where

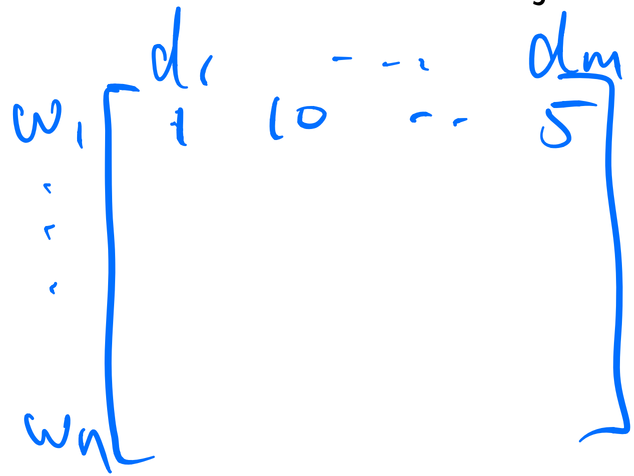
- Row and columns represent two sets of objects (e.g. words)
- Each entry is the (adjusted) co-occurence counts of two objects

Example:

cols rows

- words × documents

- word × word
- person × movie
- note × song



Reweight counts

Upweight informative words

Pointwise mutual information (PMI)

$$\text{PMI}(x; y) = \log \frac{p(x, y)}{p(x)p(y)} = \log \frac{p(x \mid y)}{p(x)} = \log \frac{p(y \mid x)}{p(y)}$$

=1

- Symmetric: $\text{PMI}(x; y) = \text{PMI}(y; x)$
- Range: $(-\infty, \min(-\log p(x), -\log p(y)))$

  $p(x|y) = 1$ or $p(y|x) = 1$
- Estimates:

$$\hat{p}(x \mid y) = \frac{\text{count}(x, y)}{\text{count}(y)} \quad \hat{p}(x) = \frac{\text{count}(x)}{\sum_{x' \in \mathcal{X}} \text{count}(x')}$$

$= \sum_x \text{count}(x, y)$

- $\text{PPMI}(x; y) \stackrel{\text{def}}{=} \max(0, \text{PMI}(x; y))$

Dimensionality reduction

Motivation: want a lower-dimensional, dense representation for efficiency

Recall **SVD**: given a $m \times n$ matrix $A_{m\times n}$ (→ word-doc matrix), we can decompose it to

$$U_{m\times m}\Sigma_{m\times n}V^T_{n\times n} ,$$

where U and V are orthogonal matrices, and Σ is a diagonal matrix.

Interpretation: consider the largest singular value σ_1,

$A = U\Sigma V^T$

$AV = U\Sigma V^T V = U\Sigma$

$$Av_1 = \sigma_1 u_1 .$$

- u_1 is a vector in the column space of A
- u_1 is the direction where the column vectors vary the most

SVD for the word-document matrix

"importance" of a cluster

$$A = \begin{array}{c} \text{US} \\ \text{UN} \\ \text{global} \\ \text{gene} \\ \text{lab} \end{array} \overset{d_1 \; \cdots \; d_n}{\begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}}_{m \times n} = \underbrace{\begin{bmatrix} | & & | \\ u_1 & \cdots & u_m \\ | & & | \end{bmatrix}}_{\text{top-}k} \begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_n \end{bmatrix} \begin{bmatrix} - & v_1^T & - \\ & \vdots & \\ - & v_n^T & - \end{bmatrix}$$

truncated SVD

u_i: doc cluster / concept

$$u_1 = \begin{bmatrix} 0.3 \\ 0.5 \\ 0.6 \\ 0.01 \\ 0.001 \end{bmatrix} \qquad u_2 = \begin{bmatrix} 0.01 \\ 0.001 \\ 0.01 \\ 0.5 \\ 0.6 \end{bmatrix}$$

political

scientific

v_i: word clusters

u_{ij}: connection between word j and doc cluster i

Summary

Vector space models

1. Design the matrix, e.g. word $\times$ document, people $\times$ movie.
2. Reweight the raw counts, e.g. TFIDF, PMI.
3. Reduce dimensionality by (truncated) SVD.
4. Use word/person/etc. vectors in downstream tasks.

Key idea:

- Represent an object by its connection to other objects in the data.
- For NLP, the word meaning can be represented by the context it occurs in.

Table of Contents

Learning word embeddings

Goal: map each word to a vector in $\mathbb{R}^d$ such that *similar* words also have *similar* word vectors.

Can we formalize this as a prediction problem?

- Needs to be self-supervised since our data is unlabeled.
- Low error $\implies$ similar words have simliar representations.

Intuition: word guessing

- Predict a word based on its context
- Predict the context given a word

The skip-gram model

Task: given a word, predict its neighboring words within a window

The quick brown fox jumps over the lazy dog

Assume conditional independence of the context words:

$$p(w_{i-k}, \ldots, w_{i-1}, w_{i+1}, \ldots, w_{i+k} \mid w_i) = \prod_{j=i-k}^{i+k} p(w_j \mid w_i)$$

How to model $p(w_j \mid w_i)$?

The skip-gram model

Use logistic regression

weight vector for class w_j

input features

$$p(w_j \mid w_i) = \frac{\exp\left[\theta_{w_j} \cdot \phi(w_i)\right]}{\sum_{w \in \mathcal{V}} \exp\left[\theta_w \cdot \phi(w_i)\right]}$$

vector representation of w_j

vector repres. of w_i

$$= \frac{\exp\left[\phi_{\text{ctx}}(w_j) \cdot \phi_{\text{wrd}}(w_i)\right]}{\sum_{w \in \mathcal{V}} \exp\left[\phi_{\text{ctx}}(w_j) \cdot \phi_{\text{wrd}}(w_i)\right]}$$

$\phi_{\text{one-hot}}(w)$

- $\phi \colon w \mapsto A_{d \times |\mathcal{V}|} \phi_{\text{BoW}}(w)$
- In practice, ϕ is implemented as a dictionary
- Learn parameters by MLE and SGD (is the objective convex?)
- ϕ_{wrd} is taken as the word embedding

The continuous bag-of-words model

Task: given the context, predict the word in the middle

The quick brown fox jumps over the lazy dog

Similary, we can use logistic regression for the prediction

$$p(w_i \mid w_{i-k}, \ldots, w_{i-1}, w_{i+1}, \ldots, w_{i+k})$$

input

How to represent the context (input feature)?

The continuous bag-of-words model

$$c = w_{i-k}, \ldots, w_{i-1}, w_{i+1}, \ldots, w_{i+k}$$

weight input

$$p(w_i \mid c) = \frac{\exp\left[\theta_{w_i} \cdot \phi_{\text{BoW}}(c)\right]}{\sum_{w \in \mathcal{V}} \exp\left[\theta_w \cdot \phi_{\text{BoW}}(c)\right]}$$

$$= \frac{\exp\left[\phi_{\text{wrd}}(w_i) \cdot \sum_{w' \in c} \phi_{\text{ctx}}(w')\right]}{\sum_{w \in \mathcal{V}} \exp\left[\phi_{\text{wrd}}(w) \cdot \sum_{w' \in c} \phi_{\text{ctx}}(w')\right]}$$

- Implementation is similar to the skip-gram model.

Properties of word embeddings

- Find synonyms
- Solve word analogy problems

  man : woman :: king : queen

  $$\phi_{\text{wrd}}(\text{man}) - \phi_{\text{wrd}}(\text{woman}) \approx \phi_{\text{wrd}}(\text{king}) - \phi_{\text{wrd}}(\text{queen})$$

  man : woman :: king : ?

  $$\arg\max_{w \in \mathcal{V}} \text{sim}(-\phi_{\text{wrd}}(\text{man}) + \phi_{\text{wrd}}(\text{woman}) + \phi_{\text{wrd}}(\text{king}), w)$$

[demo]

Comparison

vector space models	word embeddings
matrix factorization	prediction problem
fast to train	slow (with large corpus) but more flexible
interpretable components	hard to interprete but has intriguing properties

Both uses the distributional hypothesis.

Summary

Key idea: formalize word representation learning as a self-supervised prediction problem

Prediction problems:

- CBOW: Predict word from context
- Skip-gram: Predict context from words
- Other possibilities:
  - Predict $\log \hat{p}(\text{word} \mid \text{context})$, e.g. GloVe
  - Contextual word embeddings

Similar ideas can be used to learn embeddings of other objects, e.g. image, product etc.

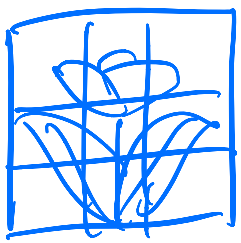
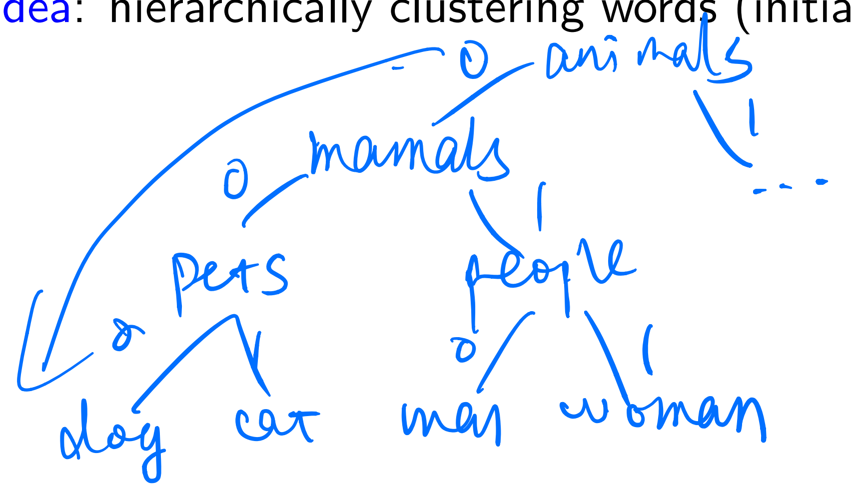


Table of Contents

Brown clustering

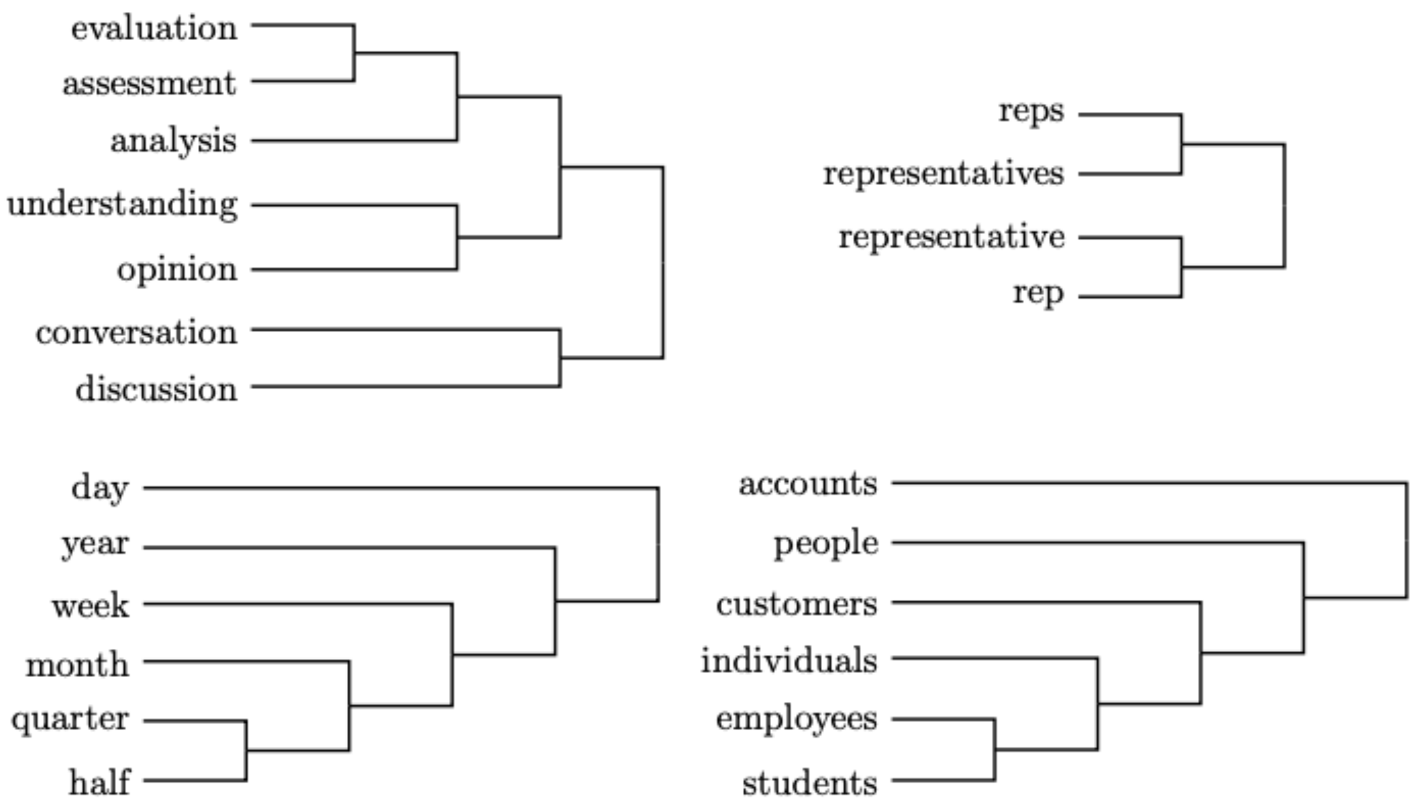
Developed at IBM by Peter Brown et al. in early 90s.

Idea: hierarchically clustering words (initially used for language modeling)

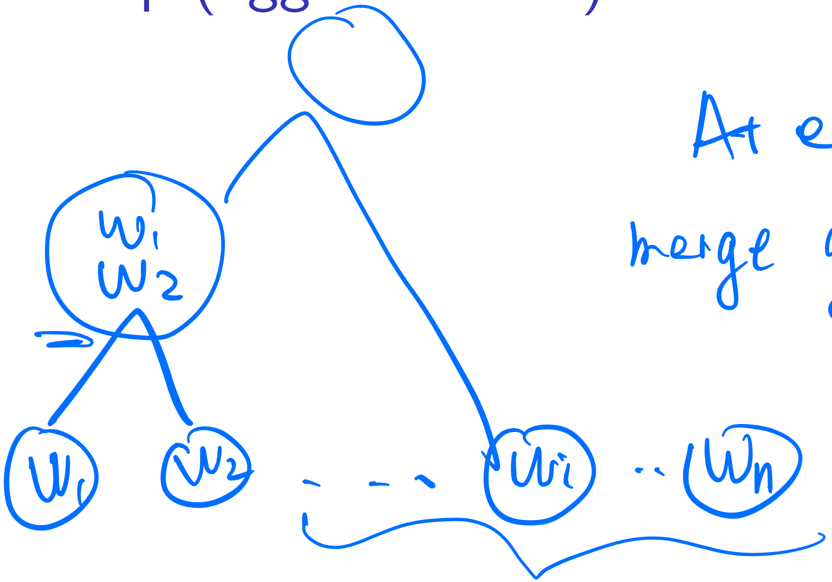


dog: [pets, manals, animals] = [0, 0, 0]

Example clusters



Bottom-up (agglomerative) clustering



At each step,

merge $\arg\max_{c_1, c_2} \text{sim}(c_1, c_2)$

where

$c_1, c_2 \in$ root nodes

$$\text{sim}(c_1, c_2) = \sum_{w_1 \in c_1} \sum_{w_2 \in c_2} s(w_1, w_2)\, \hat{p}(w_1, w_2)$$

$s \approx$ PMI

Summary

Brown clustering

1. Obtain initial word representation
2. Defind distance function between two clusters
3. Run heirarchical clustering
4. Use the (binary) "path" to a word as additional features in downstream tasks

Evaluate word vectors

Intrinsic evaluation

- Evaluate on the proxy task (related to the learning objective)
- Word similarity/analogy datasets
- Human evaluation of word clusters

Extrinsic evaluation

- Evaluate on the real/downstream task we care about
- Use word vectors as features in NER, parsing etc.

Table of Contents

Feature learning

Linear predictor with handcrafted features: $f(x) = w \cdot \phi(x)$.

Can we learn intermediate features?

Example:

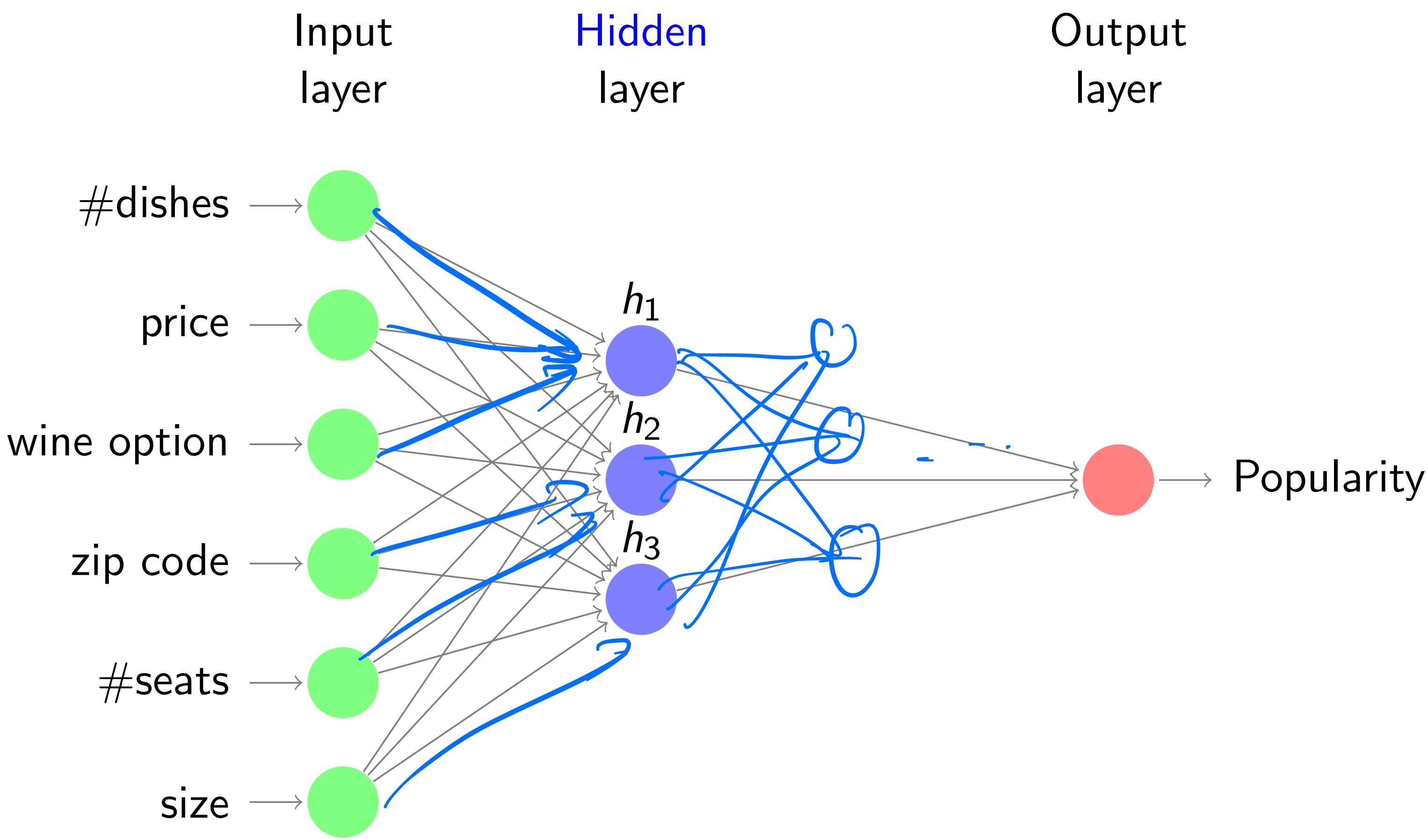
- Predict popularity of restaurants.
- Raw input: #dishes, price, wine option, zip code, #seats, size
- Decompose into subproblems:

$$h_1([\#\text{dishes, price, wine option}]) = \text{food quality}$$

$$h_2([\text{zip code}]) = \text{walkable}$$

$$h_3([\#\text{seats, size}]) = \text{nosie}$$

Learning intermediate features



Neural networks

Key idea: automatically learn the intermediate features.

Feature engineering: Manually specify $\phi(x)$ based on domain knowledge and learn the weights:

$$f(x) = w^T \phi(x).$$

Multilayer NN

$f(x) = (g \circ h^L \circ h^{L-1} \cdots \circ h^1)(x)$

Feature learning: Automatically learn both the features (K hidden units) and the weights:

$$h(x) = [h_1(x), \ldots, h_K(x)], \quad f(x) = w^T h(x)$$

output, hidden

$\sum_i w_i \sigma(v_i^T x)$ ← input

Parametrize h: $x \mapsto \sigma(v_i^T x)$.

↑ activation function, tanh, Relu $\max(0, x)$