Neural Network Models

for natural language processing

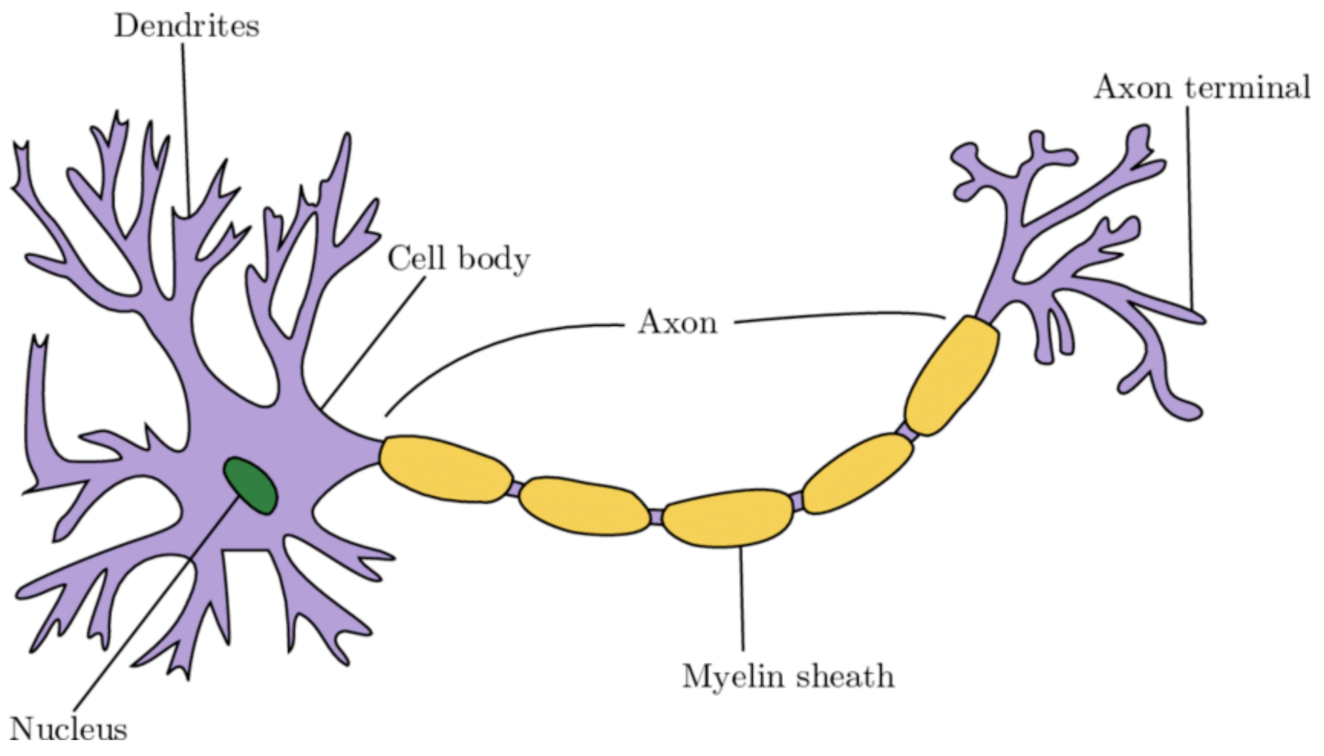
Thomas Brochhagen // Session 05, 2023 // NLP@UPF

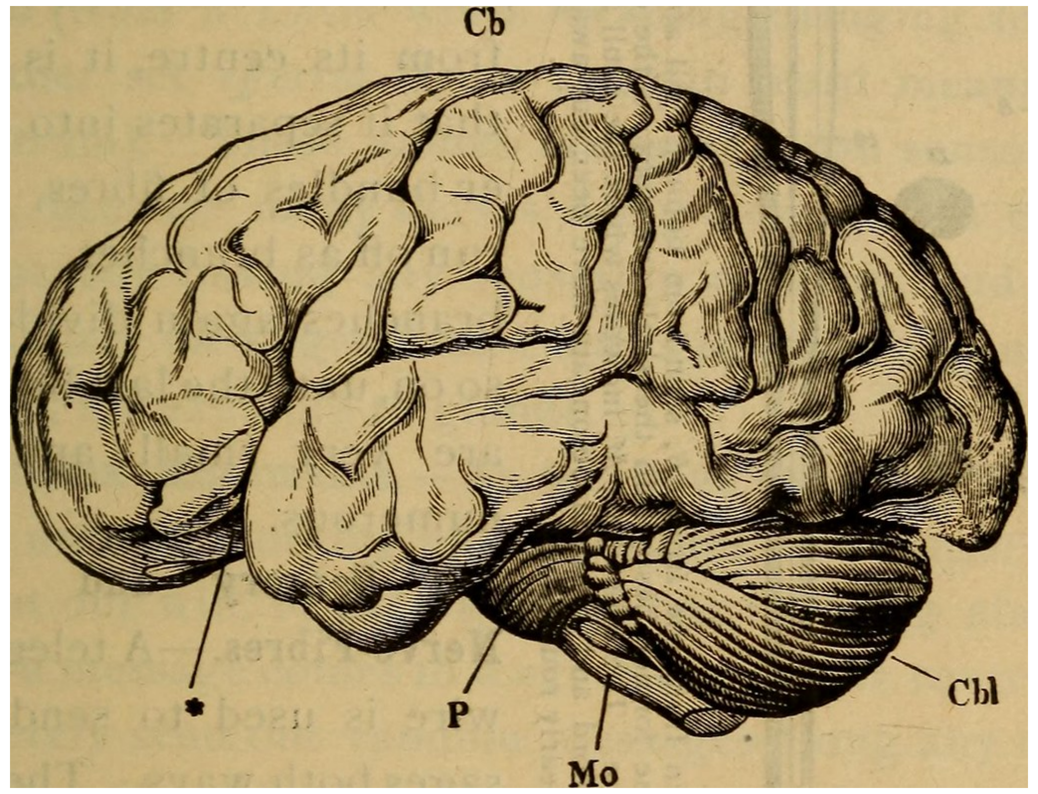
1. Linear and non-linearity models

2. Neural network models

Neural Network Model

Neural Network Model





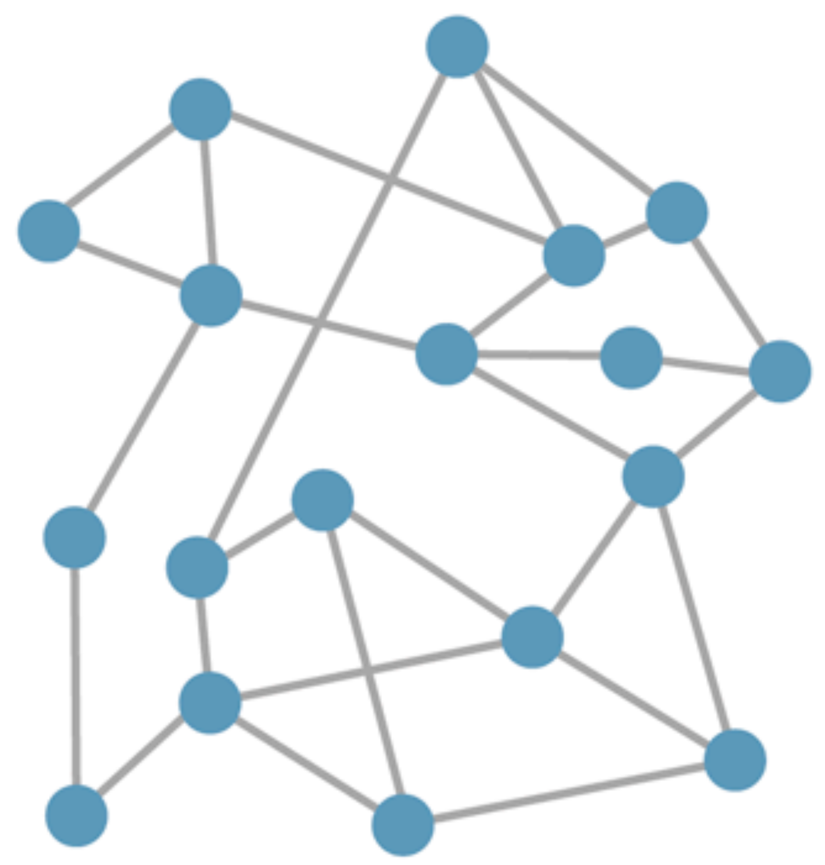
How does the brain process information / language?

Neural Network Model

$$\underbrace{\text{Neural Network}}$$

Distributed architecture
Connectionist basis

Neural Network Model



What is a model?

Neural Network **Model**

$$\underbrace{\phantom{\text{Model}}}$$

Useful representation

Often but not always
an abstraction

~~Neural~~ Network Model

Models

For any word v from vocabulary of size |V|, with a single guess

1. predict its frequency

2. predict the next word

Loss/cost function

$\lambda(y, \hat{y}) = (y - \hat{y})^2$ quadratic loss

$\lambda(y, \hat{y}) = \max(0, 1 - y \cdot \hat{y})$ hinge loss

$\log p(y \mid \theta)$ log predictive density

Loss/cost function

$\lambda(y, \hat{y}) = (y - \hat{y})^2$ quadratic loss

$\lambda(y, \hat{y}) = \max(0, 1 - y \cdot \hat{y})$ hinge loss

$\log p(y \mid \theta)$ log predictive density

To be a good model, you need to know what you will be evaluated on!

Loss/cost function

$\lambda(y, \hat{y}) = (y - \hat{y})^2$ quadratic loss

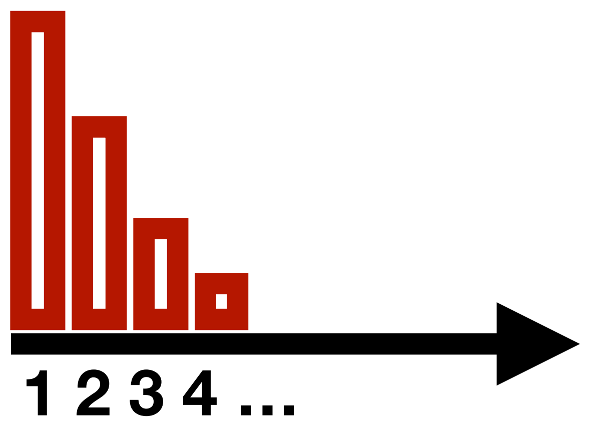
$\lambda(y, \hat{y}) = \max(0, 1 - y \cdot \hat{y})$ hinge loss

$\log p(y \mid \theta)$ log predictive density

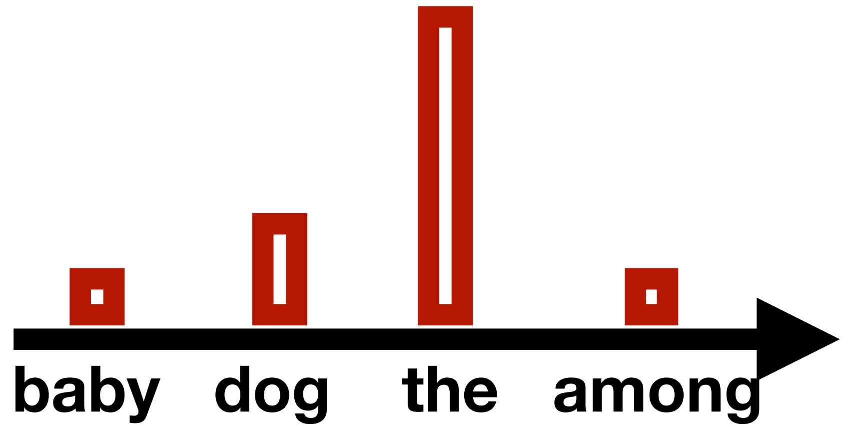
There is no free lunch

For any word v from vocabulary of size |V|, with a single guess

1. predict its frequency



2. predict the next word

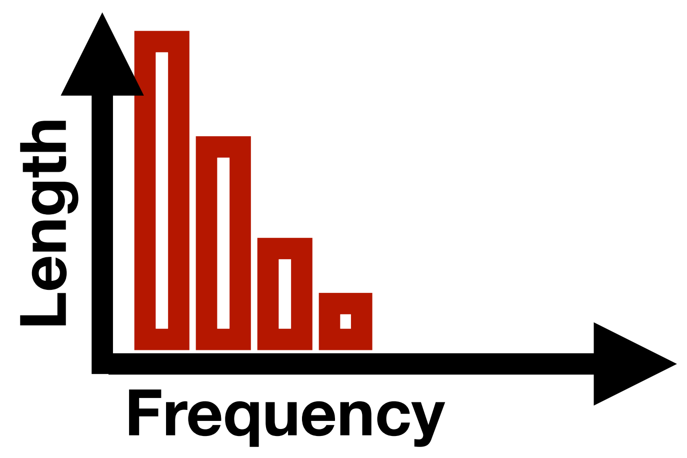


For any word v from vocabulary of size |V|, with a single **predictor**, < |V|

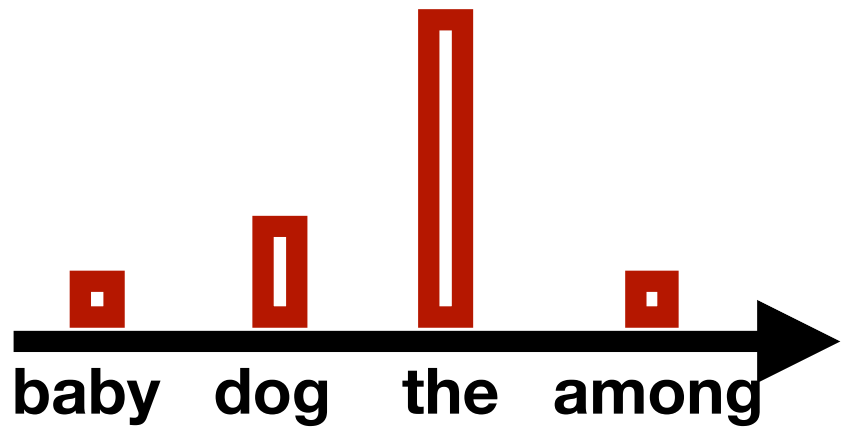
1. predict its frequency
2. predict the next word

For any word v from vocabulary of size |V|, with a single predictor, < |V|

1. predict its frequency



2. predict the next word



For any word v from vocabulary of size |V|, with a multiple predictors, < |V|

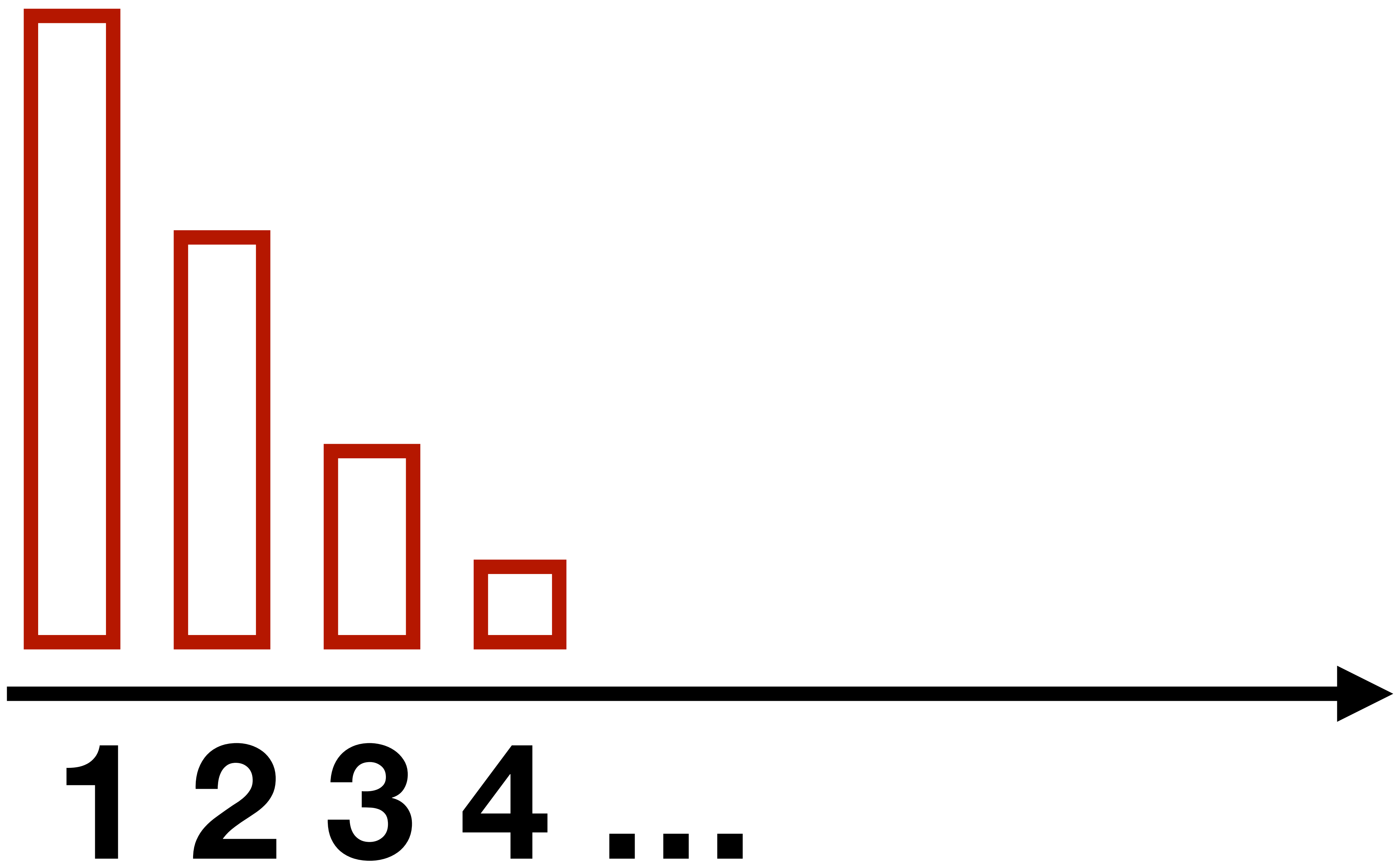
1. predict its frequency

2. predict the next word

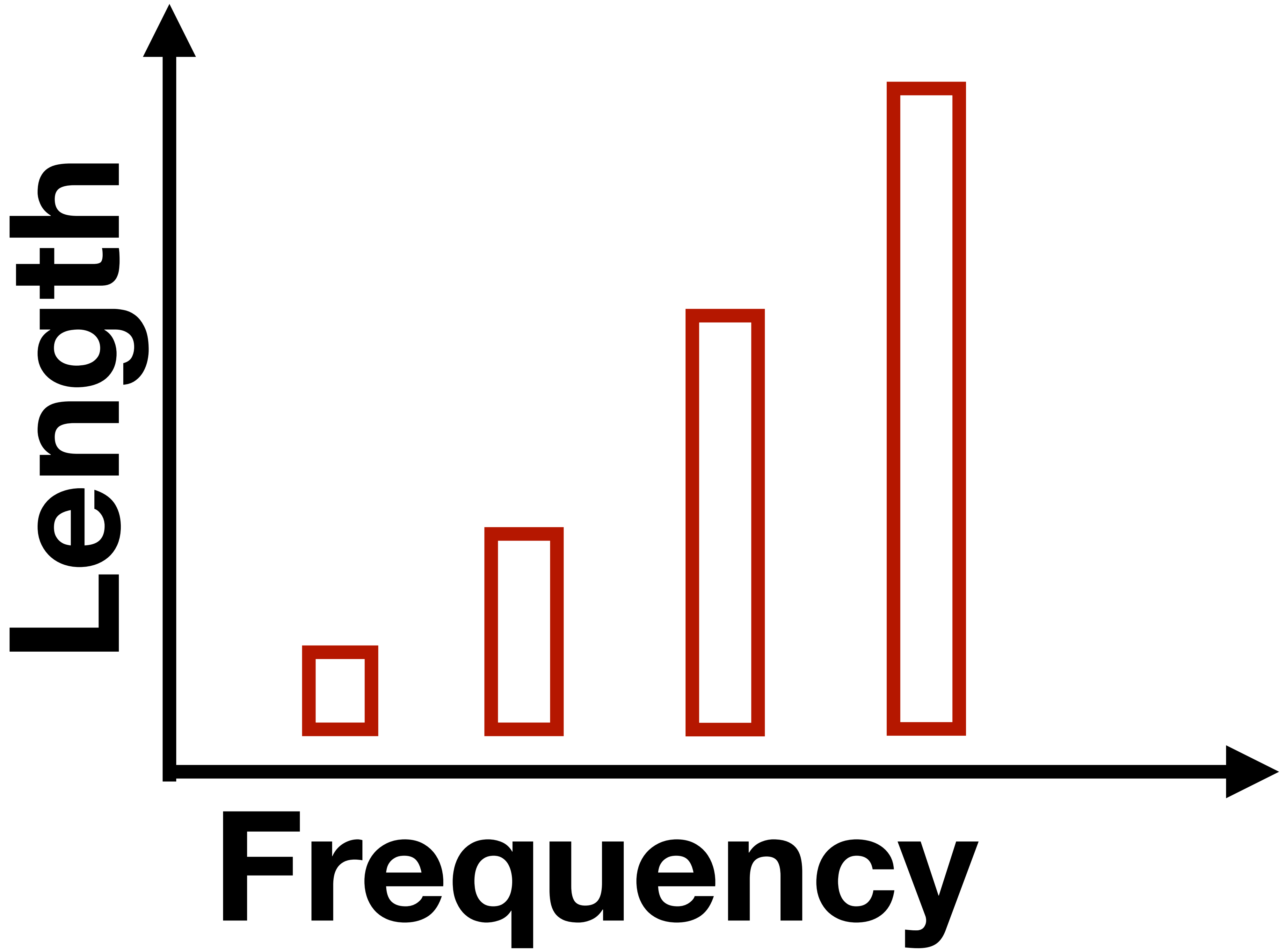
How can the multiple predictors be combined?

Linear models and non-linearity

$$\text{frequency}_i = \beta_0$$



$$\text{frequency}_i = \beta_0 + \beta_1 \text{length}_i$$



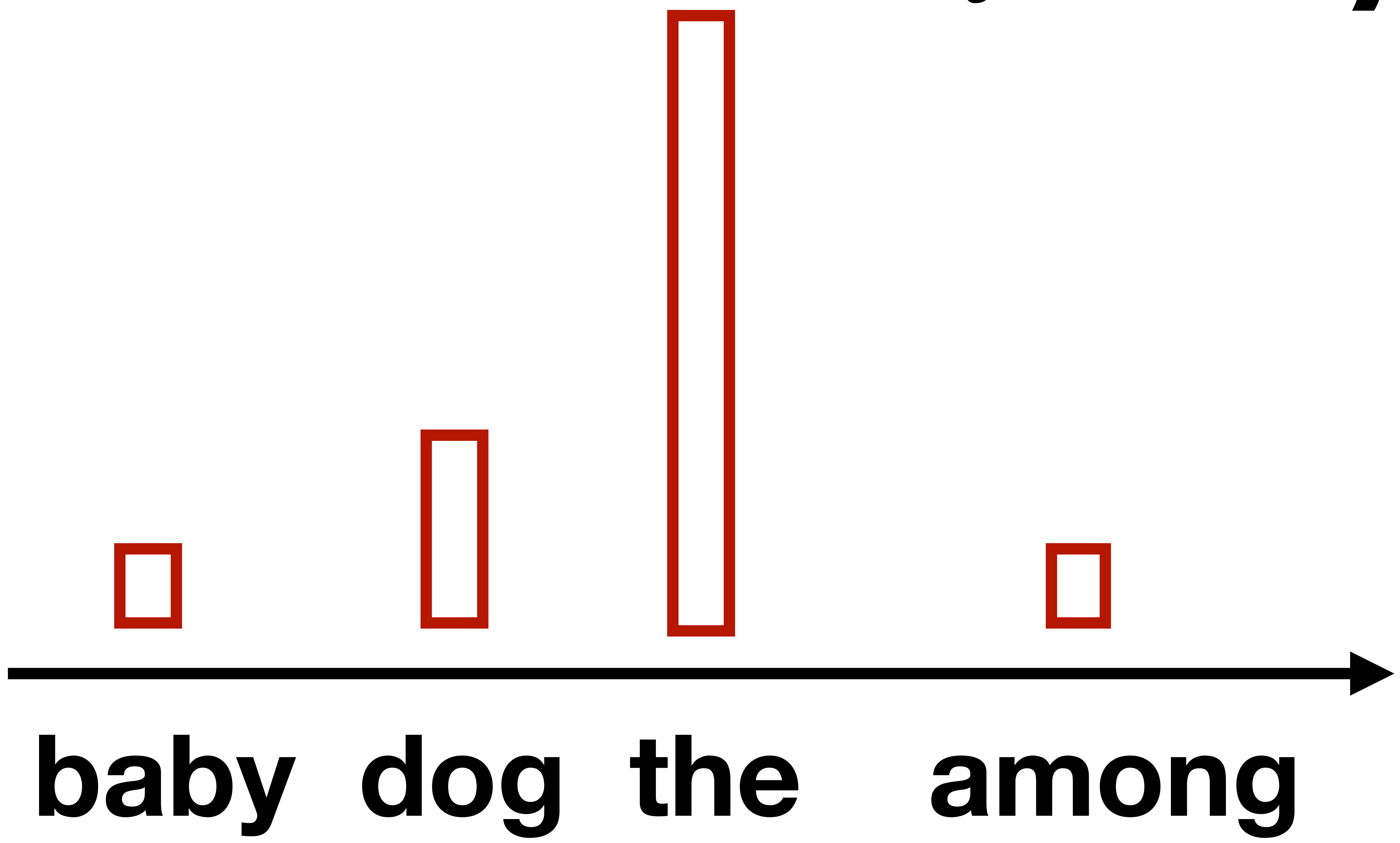
$$\text{frequency}_i = \beta_0 + \beta_1 \text{length}_i + \beta_2 \text{pos}_i$$

$$\text{frequency}_i = \beta_0 + \beta_1 \text{length}_i \times \beta_2 \text{pos}_i$$

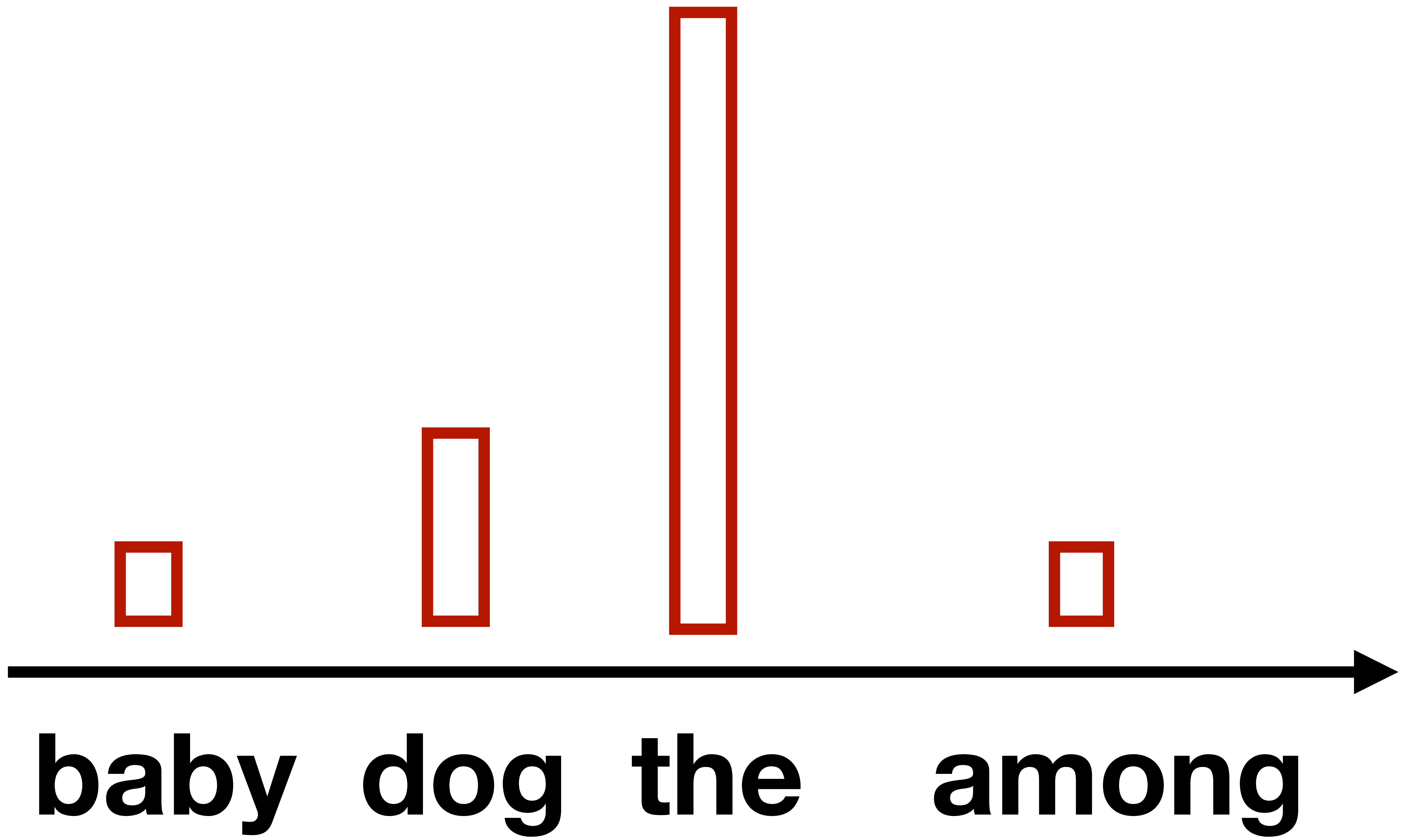
$$\cdots$$

You can keep making this more complex. The point is that this the outcome of a linear combination of parameters

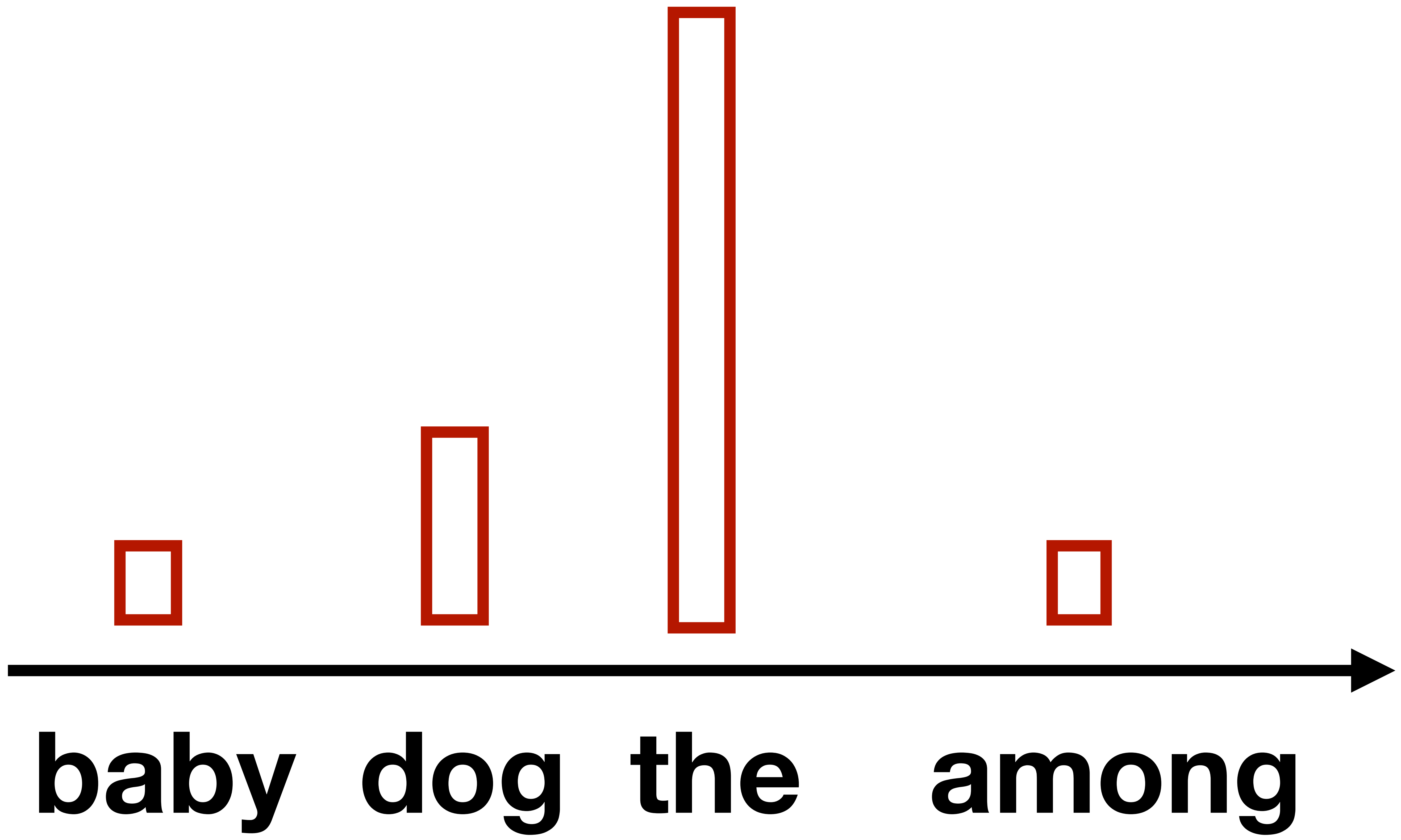
$$\text{next word}_i = \beta_0$$

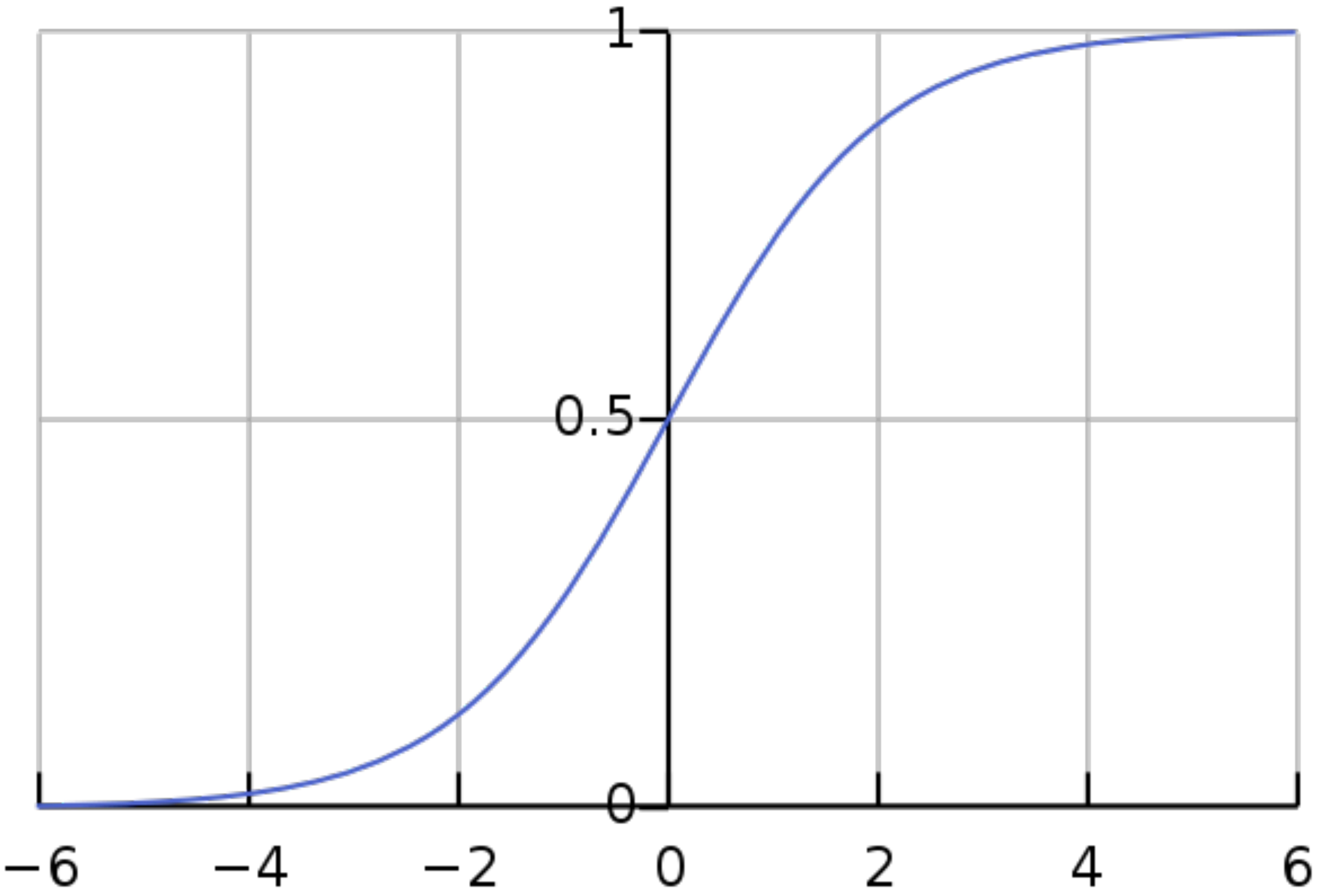


$$\text{pr}(\text{next word}_i) = \beta_0$$



$$\text{pr}(\text{next word}_i) = f(\beta_0)$$





Non-linearity, at last!

Generalized linear models are still linear models even though they use a non-linear transformation

Generalized linear models are still linear models even though they use a non-linear transformation

They explicitly estimate the effect of one or more predictors on an outcome

Generalized linear models are still linear models even though they use a non-linear transformation

They explicitly estimate the effect of one or more predictors on an outcome

NNs scale up these ideas but are non-linear and have many parameters with no clear semantics behind them

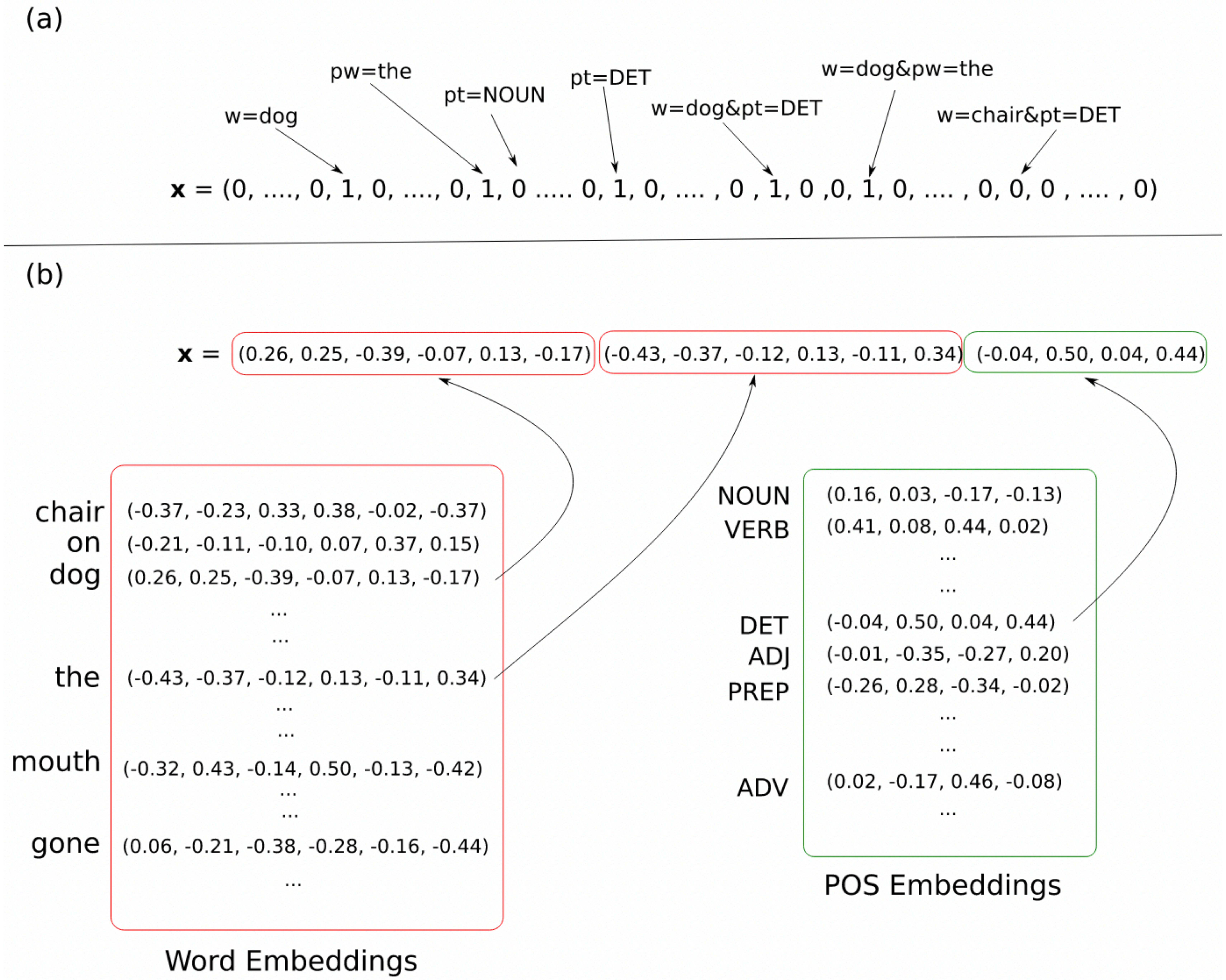
Neural Network models

Non-linearity: Many phenomena are non-linear, so linearity is a potentially unnecessary constraint in the relationship between input and output

Parameters with no clear semantics: Automatically induced from data with no need to match architecture to phenomenon*

Many parameters: Can be an issue but doesn't need to be

Dense representations



(a)
w=dog
pw=the
pt=NOUN
pt=DET
w=dog&pw=the
w=dog&pt=DET
w=chair&pt=DET
x = (0, ..., 0, 1, 0, ..., 0, 1, 0 ..... 0, 1, 0, ... , 0 , 1, 0 ,0, 1, 0, ... , 0, 0, 0 , ... , 0)
(b)
x = (0.26, 0.25, -0.39, -0.07, 0.13, -0.17) (-0.43, -0.37, -0.12, 0.13, -0.11, 0.34) (-0.04, 0.50, 0.04, 0.44)
chair (-0.37, -0.23, 0.33, 0.38, -0.02, -0.37)
on (-0.21, -0.11, -0.10, 0.07, 0.37, 0.15)
dog (0.26, 0.25, -0.39, -0.07, 0.13, -0.17)
the (-0.43, -0.37, -0.12, 0.13, -0.11, 0.34)
mouth (-0.32, 0.43, -0.14, 0.50, -0.13, -0.42)
gone (0.06, -0.21, -0.38, -0.28, -0.16, -0.44)
Word Embeddings
NOUN (0.16, 0.03, -0.17, -0.13)
VERB (0.41, 0.08, 0.44, 0.02)
DET (-0.04, 0.50, 0.04, 0.44)
ADJ (-0.01, -0.35, -0.27, 0.20)
PREP (-0.26, 0.28, -0.34, -0.02)
ADV (0.02, -0.17, 0.46, -0.08)
POS Embeddings

What are the advantages and disadvantages of sparse vs. dense representations?

Feed-forward NN

$$[x_1, \ldots, x_{d_{in}}] \quad \Rightarrow \quad \text{NN}(\,\cdot\,) \quad \Rightarrow [y_1, \ldots, y_{d_{out}}]$$

Feed-forward NN

$$[x_1, \ldots, x_{d_{in}}] \quad \Rightarrow \quad \text{NN}(\cdot) \quad \Rightarrow [y_1, \ldots, y_{d_{out}}]$$

This assumes a fixed dimensional input, but what about cases where the features in the input are variable?

For example, document classification with each word as a feature

Continous bag of words (CBOW)

$$\text{CBOW}(f_1, \ldots, f_k) = \frac{1}{k} \sum^{i=k} v(f_i)$$

$$\text{WCBOW}(f_1, \ldots, f_k) = \frac{1}{\sum^{i=k} a_i} \sum^{i=k} a_i v(f_i)$$

Interim summary feed-forward networks

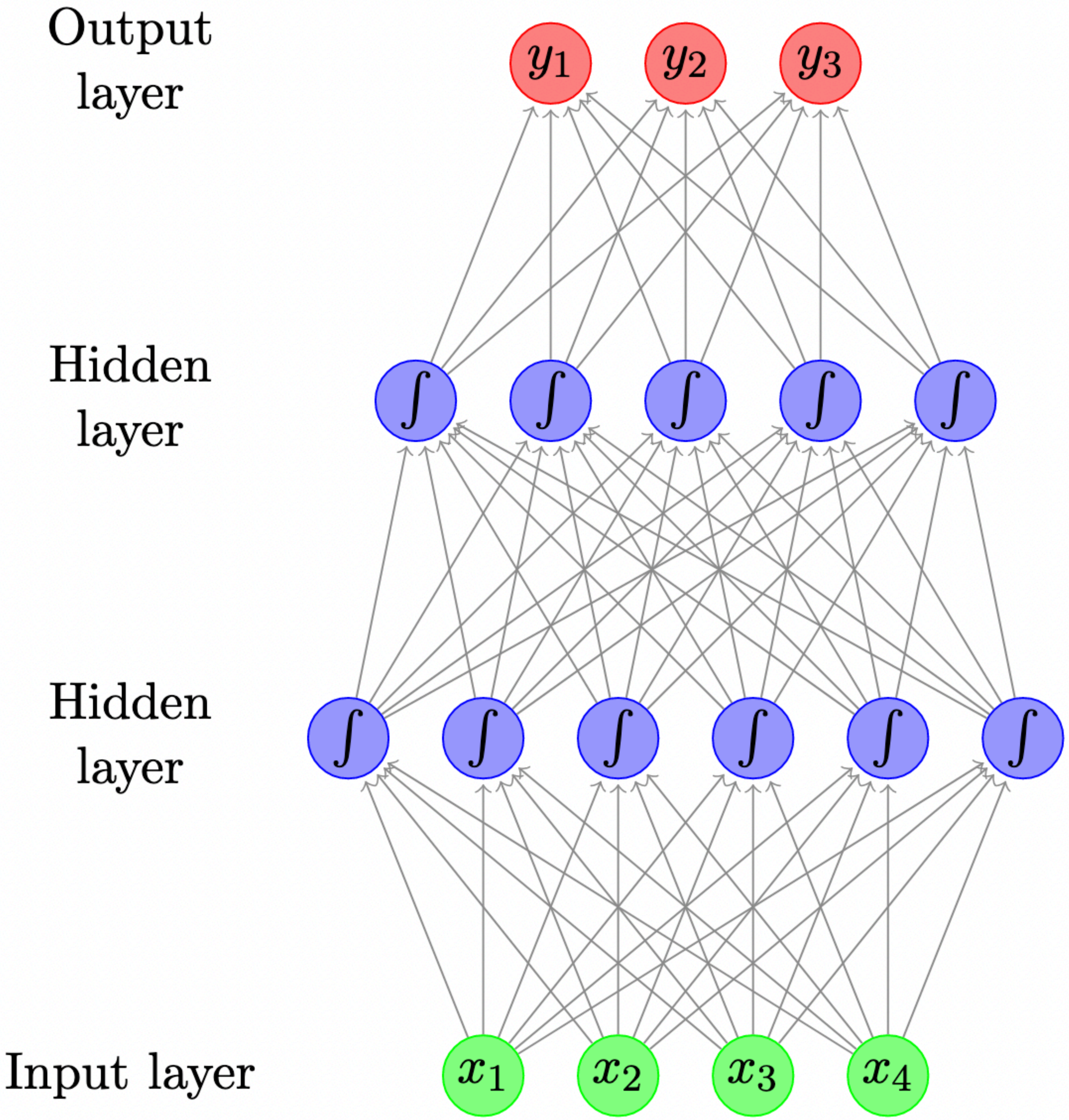
1. Extract set of core linguistic features that are relevant for predicting the output class

2. For each feature, retrieve the corresponding vector

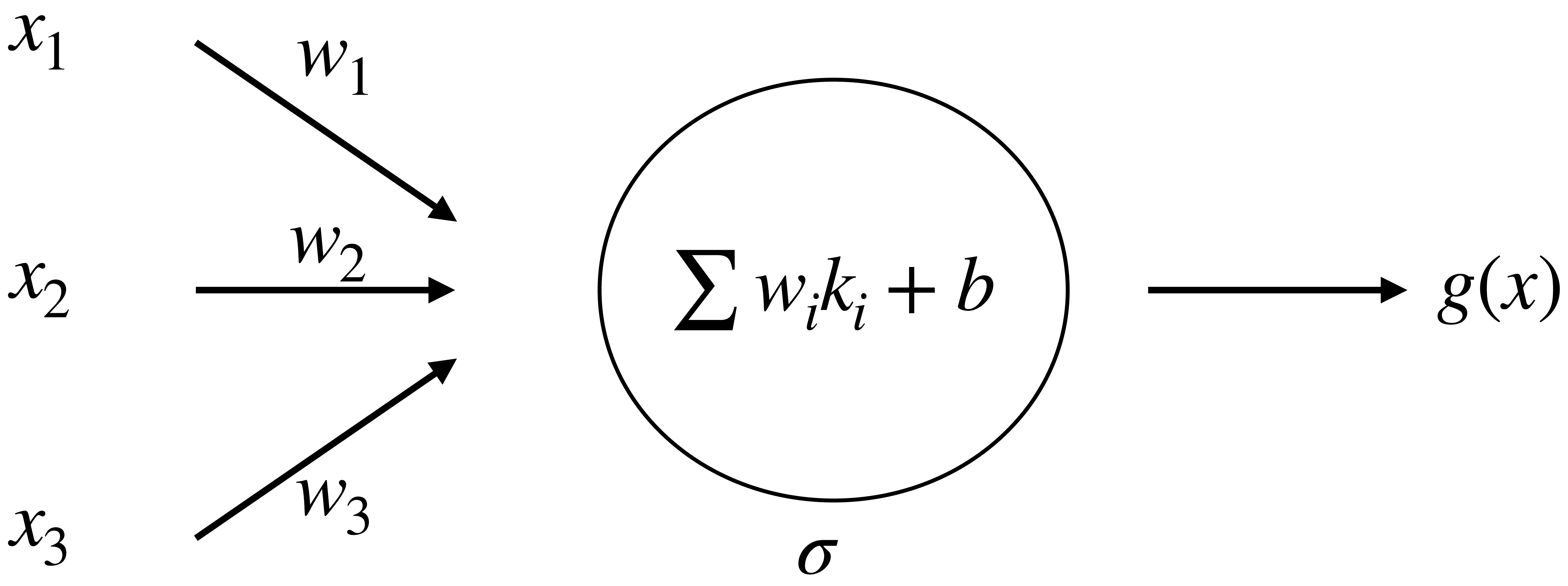
3. Combine the vectors (concatenate, sum, average, …)

4. Feed vector to the non-linear classifier: your feed-forward NN

Feed-forward NN

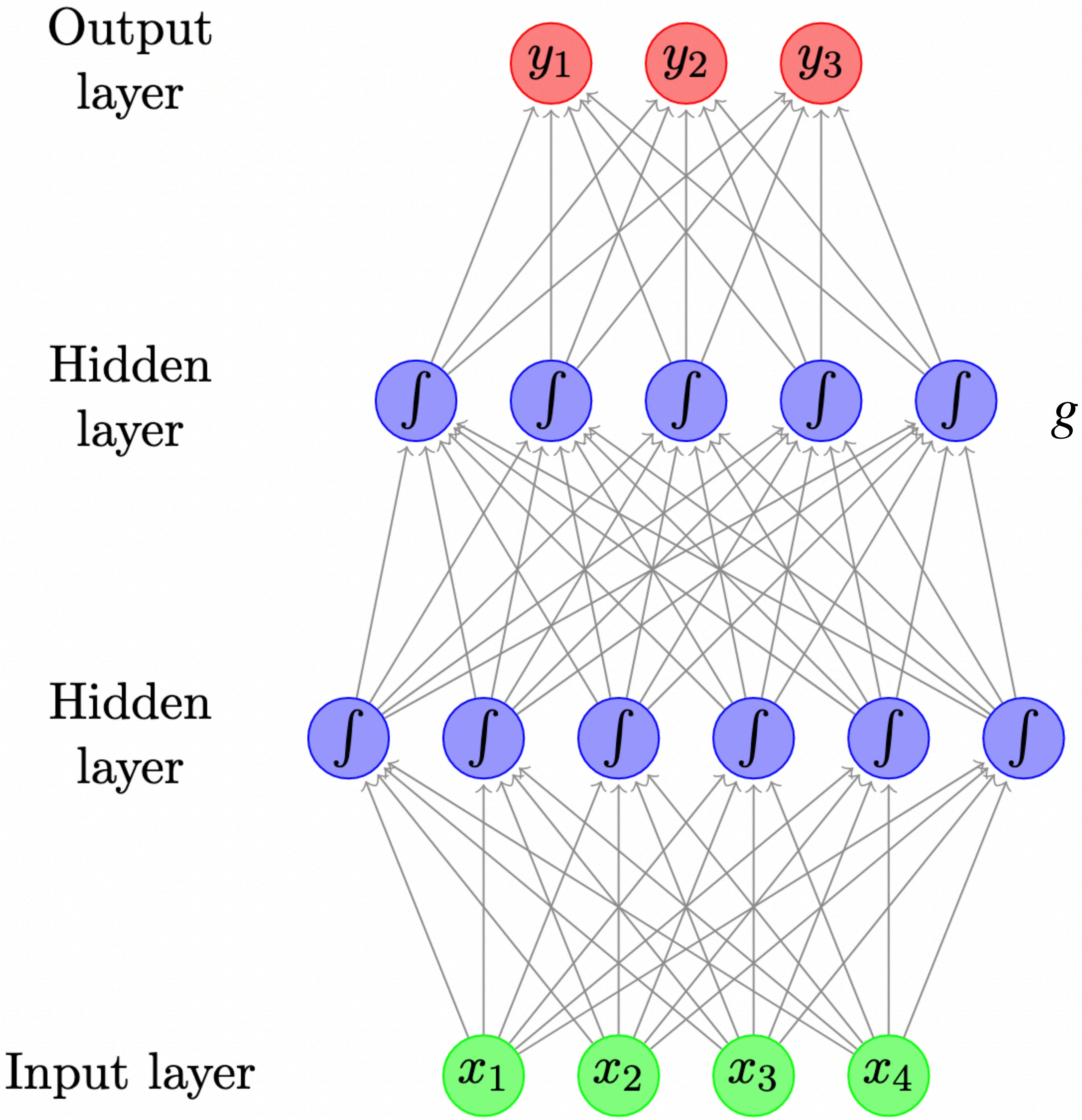


Feed-forward NN



$$\sigma(\sum_1^3 w_i k_i + b) = g(x)$$

Feed-forward NN

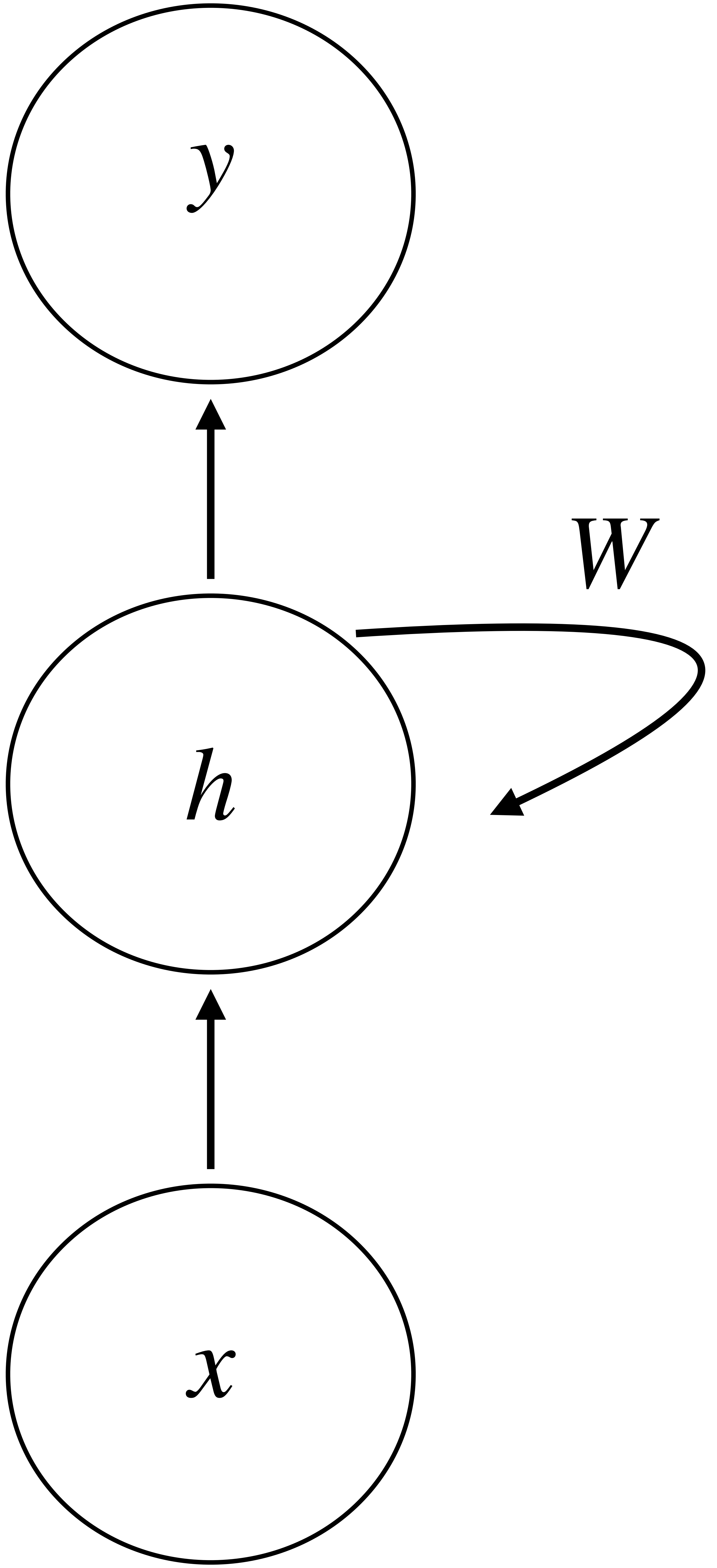


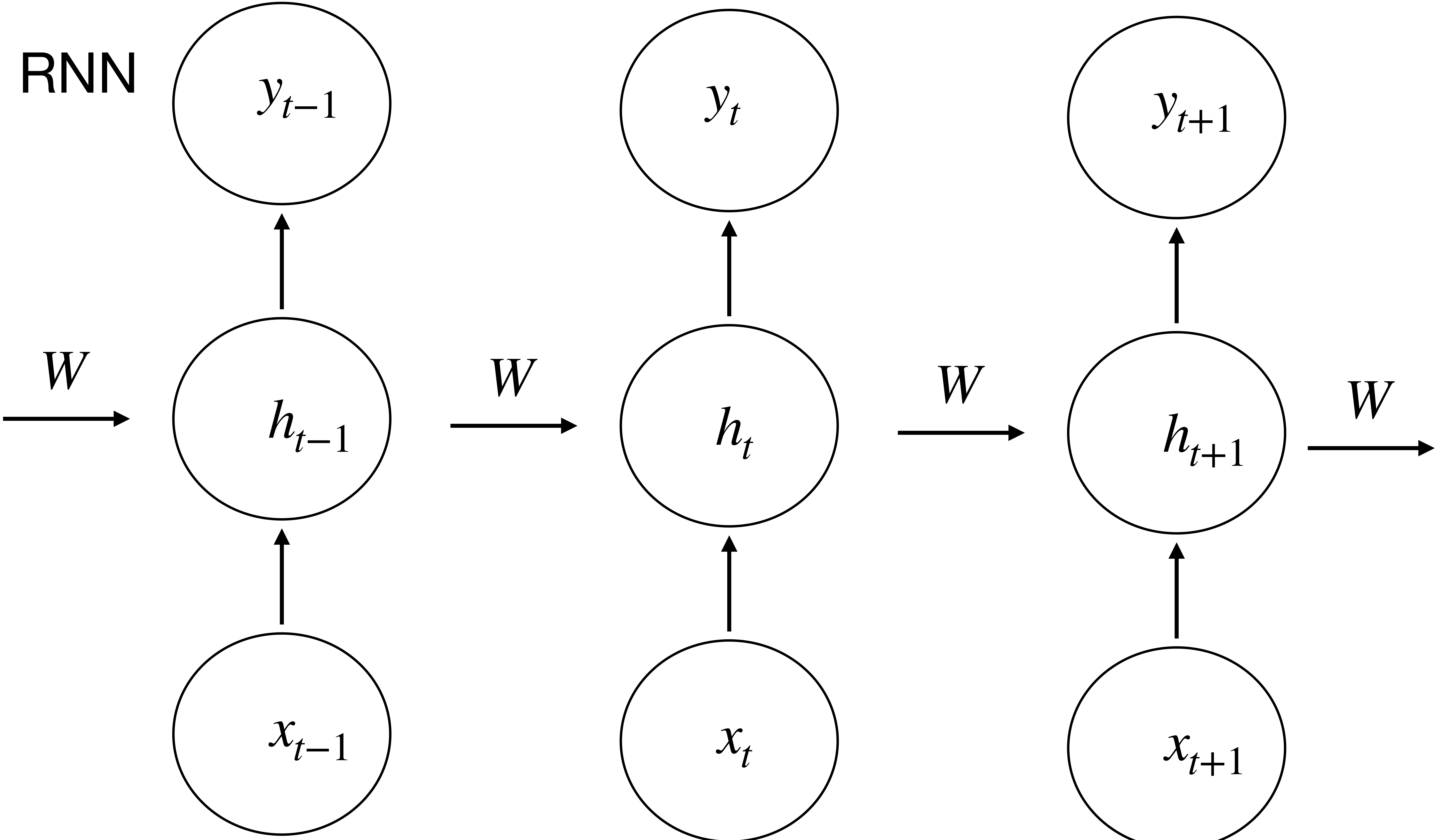
$$g^2(g^1(xW^1 + b^1)W^2 + b^2)$$

$$g^1(xW^1 + b^1)$$

$$x \in \mathbb{R}^{d_{in}}$$

RNN





RNN
y_{t-1}
y_t
y_{t+1}
W
h_{t-1}
W
h_t
W
h_{t+1}
W
x_{t-1}
x_t
x_{t+1}

Decisions for the modeller/engineer

1. Number of dimensions

Decisions for the modeller/engineer

1. Number of dimensions

2. Number of layers

Decisions for the modeller/engineer

1. Number of dimensions

2. Number of layers

3. Non-linearities (e.g., sigmoid, tanh, rectifier)

Decisions for the modeller/engineer

1. Number of dimensions

2. Number of layers

3. Non-linearities (e.g., sigmoid, tanh, rectifier)

4. Output transformation (e.g. softmax)

Decisions for the modeller/engineer

1. Number of dimensions

2. Number of layers

3. Non-linearities (e.g., sigmoid, tanh, rectifier)

4. Output transformation (e.g. softmax)

5. Connectivity

Decisions for the modeller/engineer

1. Number of dimensions

2. Number of layers

3. Non-linearities (e.g., sigmoid, tanh, rectifier)

4. Output transformation (e.g. softmax)

5. Connectivity

6. Loss function

Decisions for the modeller/engineer

1. Number of dimensions

2. Number of layers

3. Non-linearities (e.g., sigmoid, tanh, rectifier)

4. Output transformation (e.g. softmax)

5. Connectivity

6. Loss function

7. Training regime
   (e.g., stochastic gradient descent + flavor;
   batching; drop-out)

What are the main strengths and weaknesses of NNs?