CS11-737 Multilingual NLP

# Text Classification and Sequence Labeling

Lei Li

https://lileicc.github.io/course/11737mnlp23fa/

Carnegie Mellon University
Language Technologies Institute

# Text Classification

- Given an input text X, predict an output label y

Topic Classification

I like peaches and → food / politics / music / …

I like peaches and herb → food / politics / music / …

Language Identification

I like peaches and pears → English / Japanese / German / …

桃と梨が好き → English / Japanese / German / …

Sentiment Analysis (sentence/document-level)

I like peaches and pears → positive / neutral / negative

I hate peaches and pears → positive / neutral / negative

*… and many many more!*

# Sequence Labeling

- Given an input text *X*, predict an output label sequence *Y* *of equal length!*

<u>Part of Speech Tagging</u>

| He | saw | two | birds |
|---|---|---|---|
| ↓ | ↓ | ↓ | ↓ |
| PRON | VERB | NUM | NOUN |

<u>Lemmatization</u>

| He | saw | two | birds |
|---|---|---|---|
| ↓ | ↓ | ↓ | ↓ |
| he | see | two | bird |

<u>Morphological Tagging</u>

| He | saw | two | birds |
|---|---|---|---|
| ↓ | ↓ | ↓ | ↓ |
| PronType=prs | Tense=past, VerbForm=fin | NumType=card | Number=plur |

*... and more!*

# Span Labeling

- Given an input text *X*, predict an output spans and labels *Y*.

Named Entity Recognition

Leo Messi plays for Inter Miami CF

PER ORG

Syntactic Chunking

Leo Messi plays for Inter Miami CF

NP VP NP

Semantic Role Labeling

Leo Messi plays for Inter Miami CF

Agent Predicate Theme

*... and more!*

# Span Labeling as Sequence Labeling

- Predict **B**eginning, **I**n, and **O**ut tags for each word in a span

Leo Messi plays for Inter Miami CF

$\underbrace{\text{Leo Messi}}_{\text{PER}}$ plays for $\underbrace{\text{Inter Miami CF}}_{\text{ORG}}$

↓

| Leo | Messi | plays | for | Inter | Miami | CF |
|---|---|---|---|---|---|---|
| B-PER | I-PER | O | O | B-ORG | I-ORG | I-ORG |

# Text Segmentation

- Given an input text *X*, split it into segmented text *Y*.

Tokenization

A well-conceived "thought exercise."

A well - conceived " thought exercise . "

Word Segmentation (very important for web search)

南京市长江大桥

| | | | | | |
|---|---|---|---|---|---|
| 南京市 | 长江 | 大桥 | 南京 | 市长 | 江大桥 |
| Nanjing | Yangtze River | bridge | Nanjing | mayor | Jiang Da Qiao |

Morphological Segmentation

Köpekler

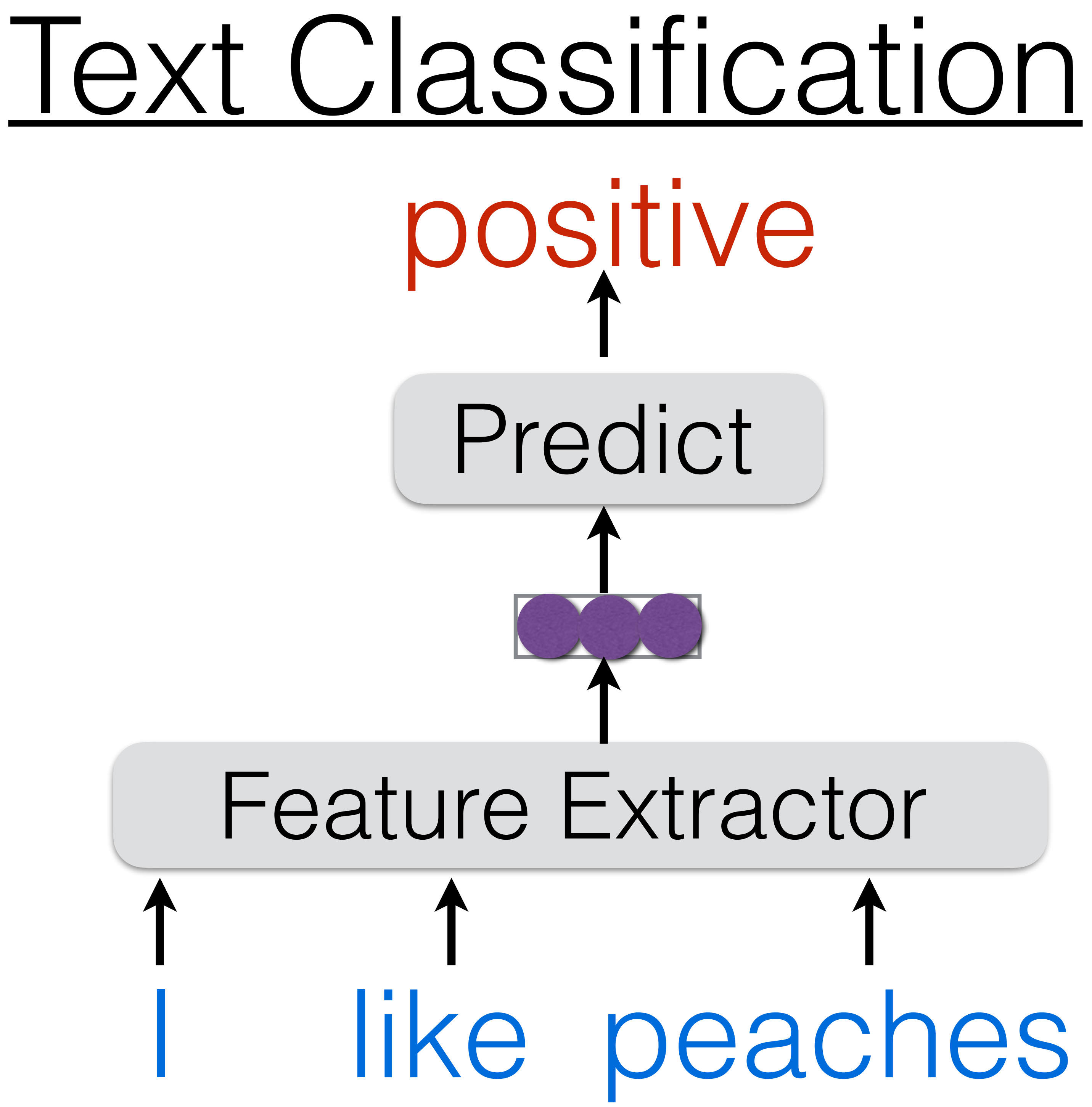
| | | | |
|---|---|---|---|
| Köpek | ler | Köpekle | r |
| dog | Number=Plural | dog_paddle | Tense=Aorist |

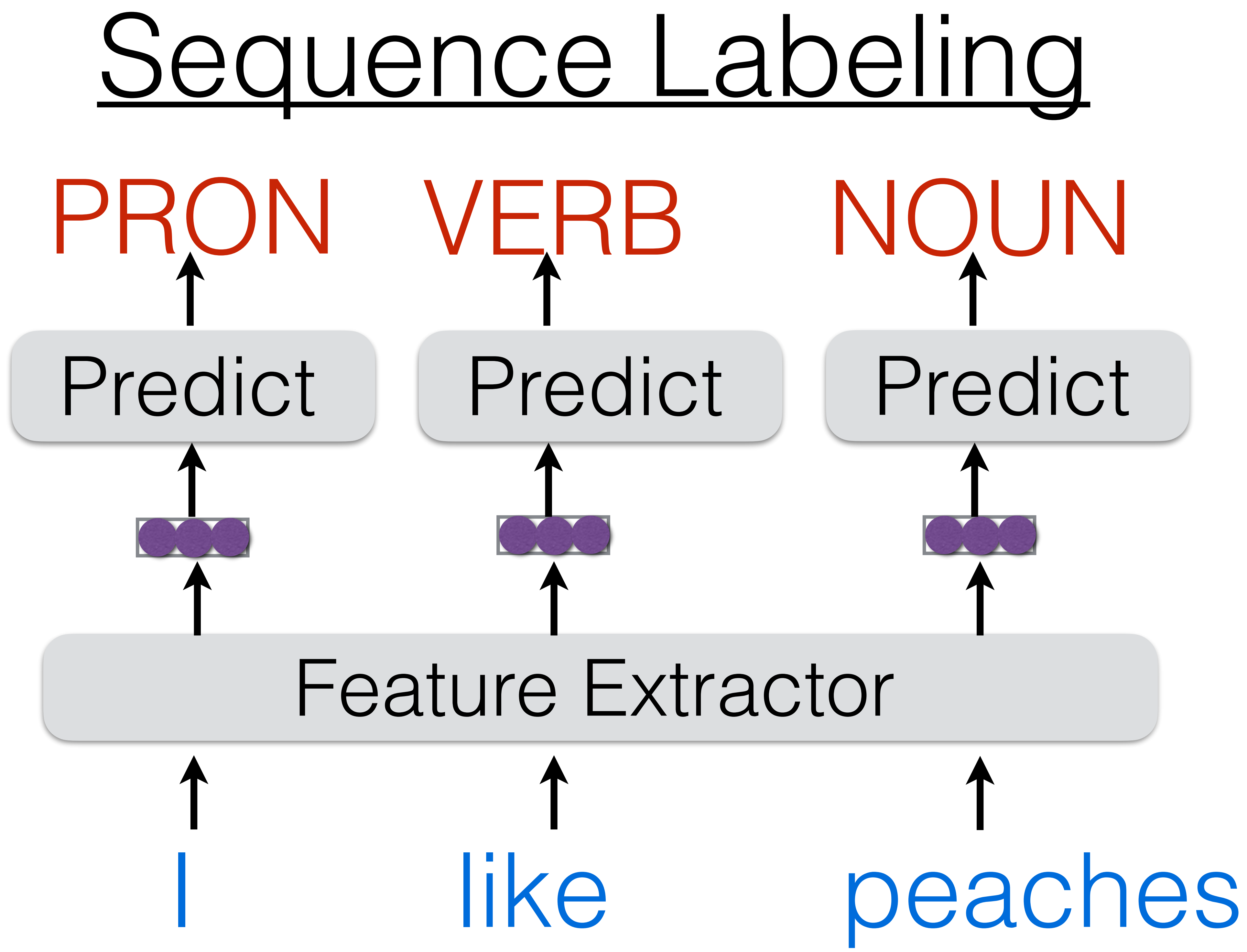
- Rule-based (statistical), or span labeling models

# Modeling for Sequence Labeling/ Classification

# How do we Make Predictions?

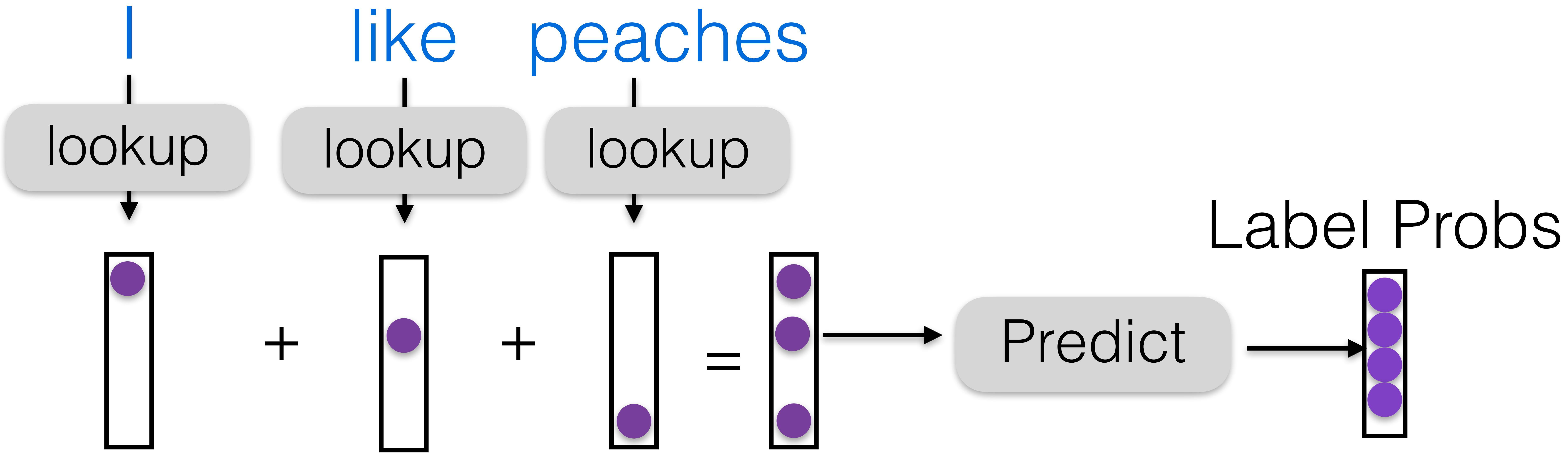
- Given an input text *X*
- Extract features *H*
- Predict labels *Y*



# A Simple Feature Extractor: Bag of Words (BOW)

- Each word has a vector of weights for each tag



# A Simple Predictor: Linear Transform+Softmax

p = softmax( W * **h** + b )

Softmax converts arbitrary scores into probabilities

$$p_i = \frac{e^{s_i}}{\sum_j e^{s_j}}$$

$$s = \begin{pmatrix} -3.2 \\ -2.9 \\ 1.0 \\ 2.2 \\ 0.6 \\ \ldots \end{pmatrix} \longrightarrow p = \begin{pmatrix} 0.002 \\ 0.003 \\ 0.329 \\ 0.444 \\ 0.090 \\ \ldots \end{pmatrix}$$

# Problem: Language is not a Bag of Words!

I don’t love pears

There’s nothing I don’t love about pears

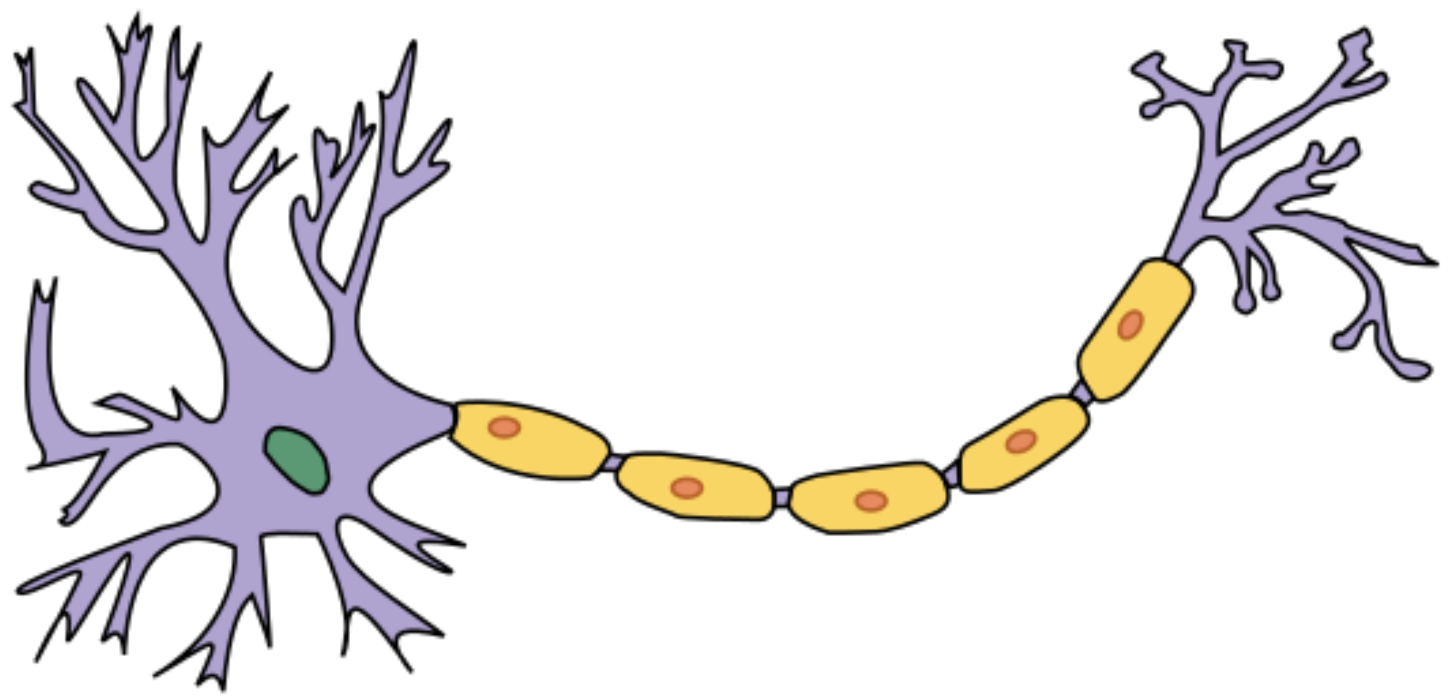
# Better Featurizers

- Bag of n-grams
- Syntax-based features (e.g. subject-object pairs)
- Neural networks
  - Recurrent neural networks
  - Convolutional networks
  - Self attention

# What is a Neural Net?: Computation Graphs

# "Neural" Nets

- Original Motivation: The Neurons in Brain



- Neural Network is a Computational graph

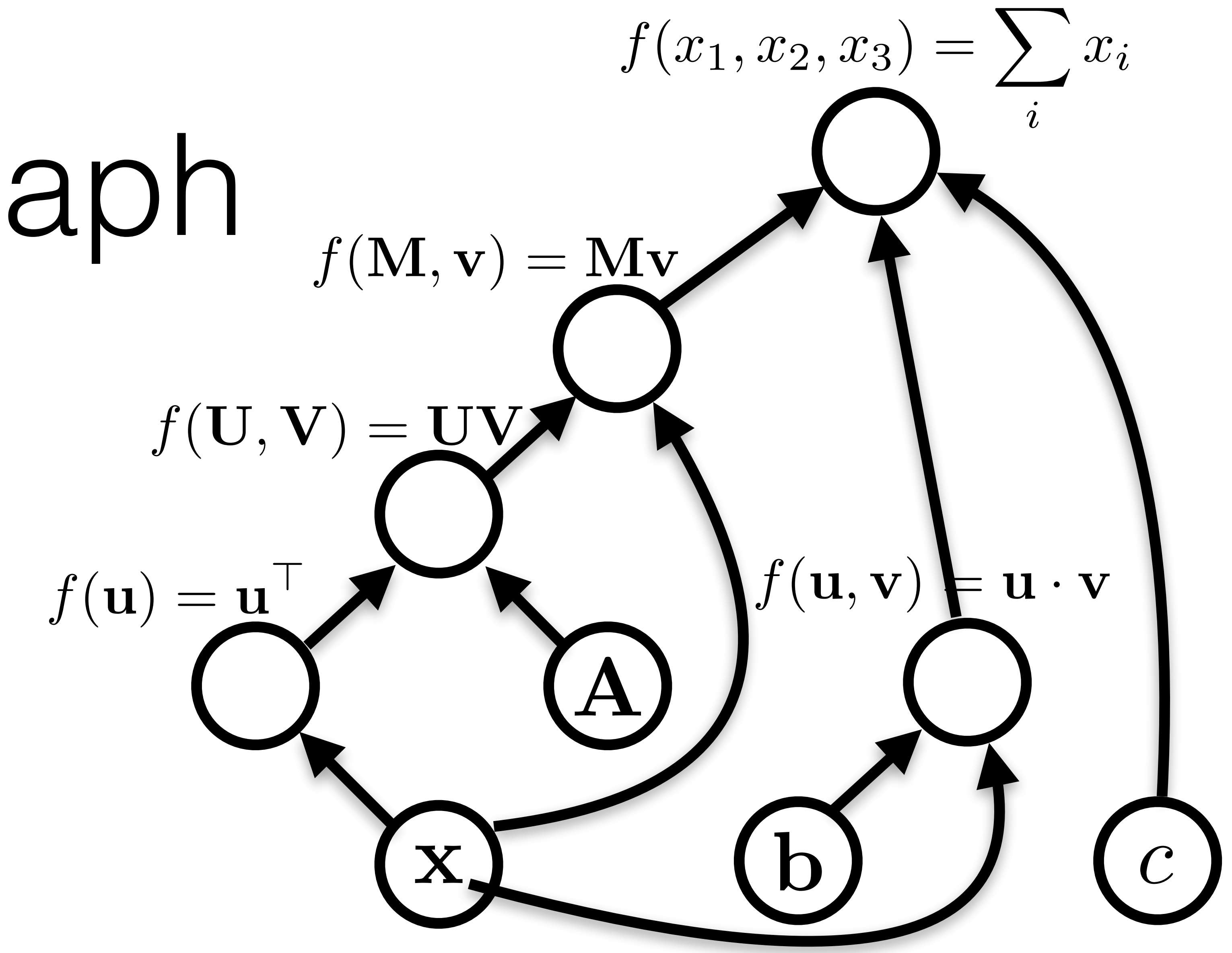



Image credit: Wikipedia

expression:

$$\mathbf{x}$$

graph:

A **node** is a {tensor, matrix, vector, scalar} value

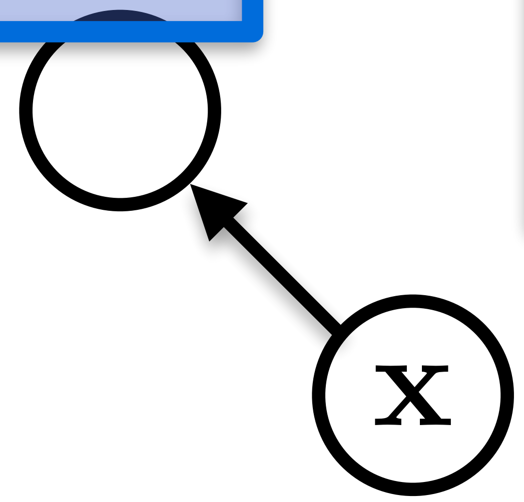
$$\mathbf{x}$$

An **edge** represents a function argument.

A **node** with an incoming **edge** is a **function** of that edge's tail node.

A **node** knows how to compute its value and the *value of its derivative w.r.t each argument (edge) times a derivative of an arbitrary input* $\frac{\partial \mathcal{F}}{\partial f(\mathbf{u})}$ .

$f(\mathbf{u}) = \mathbf{u}^\top$

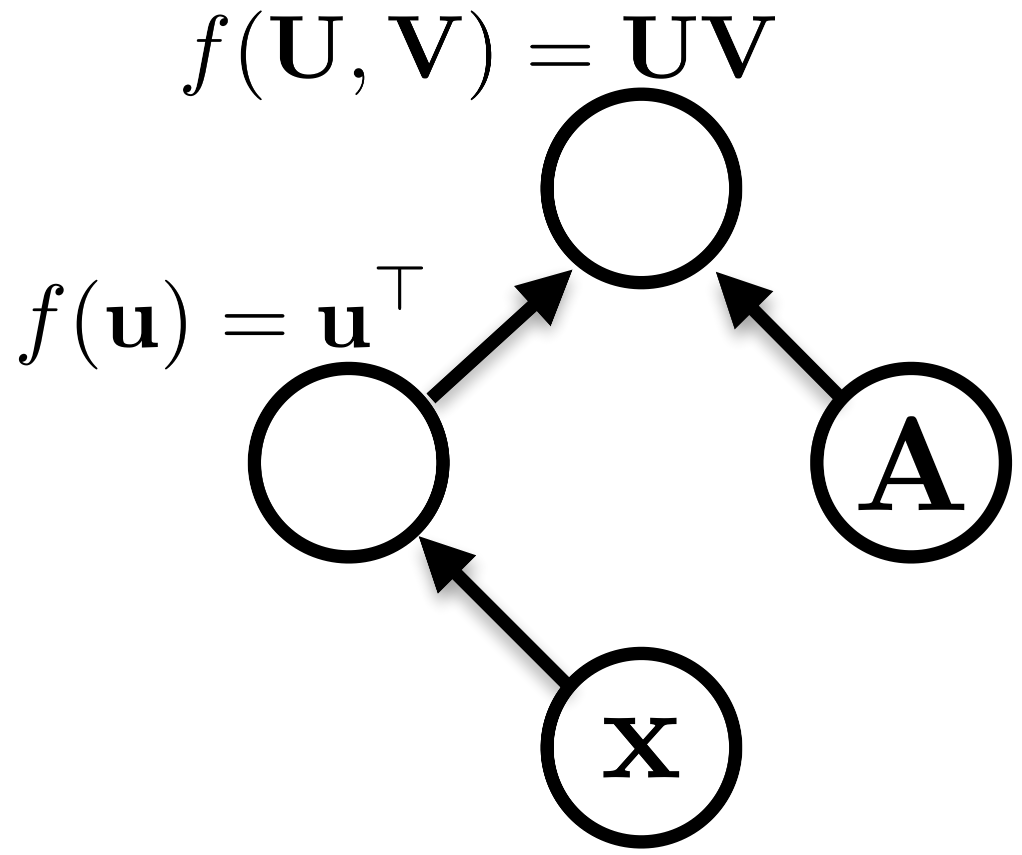



$$\frac{\partial f(\mathbf{u})}{\partial \mathbf{u}} \frac{\partial \mathcal{F}}{\partial f(\mathbf{u})} = \left(\frac{\partial \mathcal{F}}{\partial f(\mathbf{u})}\right)^\top$$

expression:

$$\mathbf{x}^\top \mathbf{A}$$

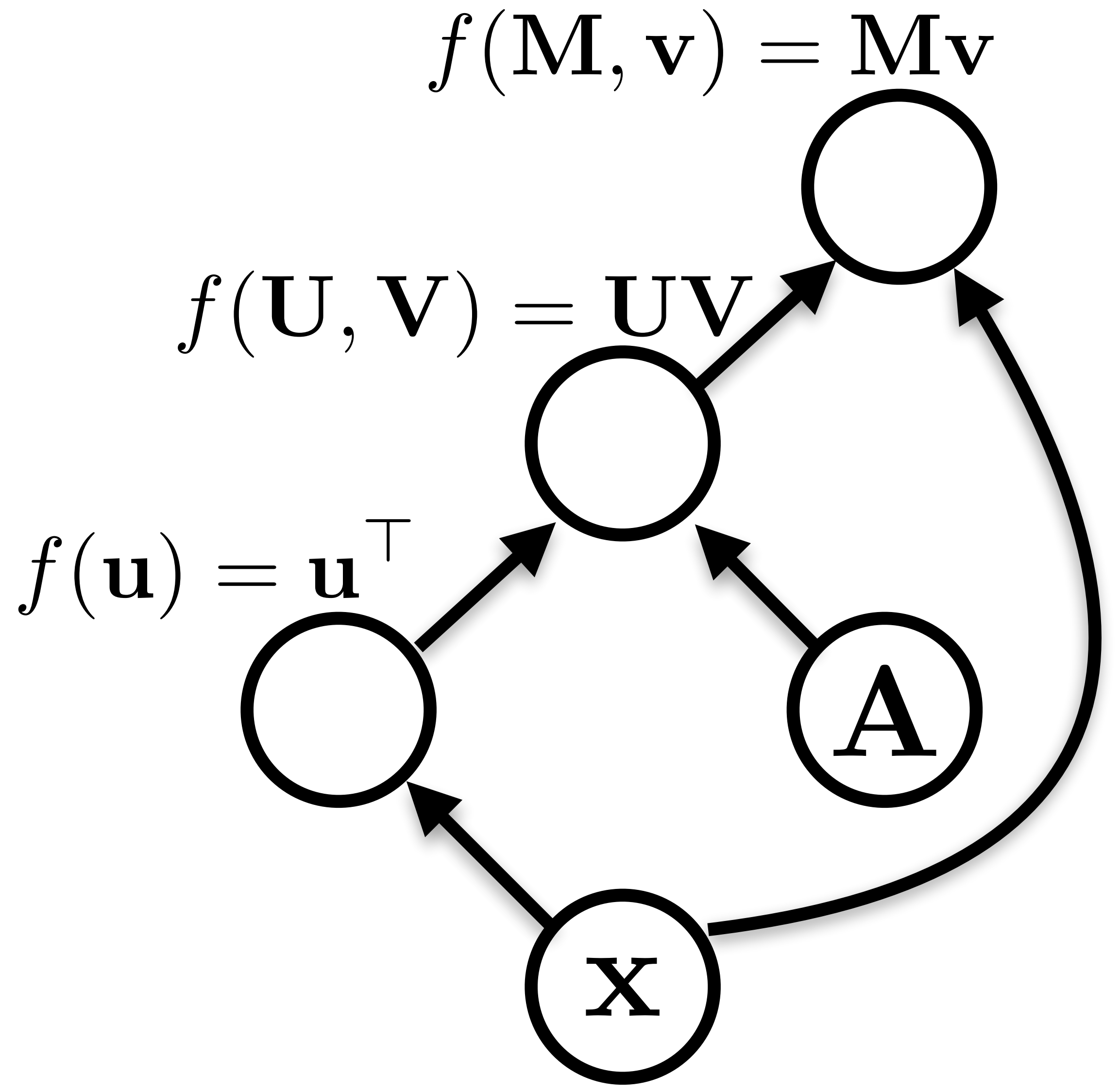
graph: Functions can be nullary, unary, binary, … *n*-ary. Often they are unary or binary.



expression:

$$\mathbf{x}^\top \mathbf{A}\mathbf{x}$$

graph:

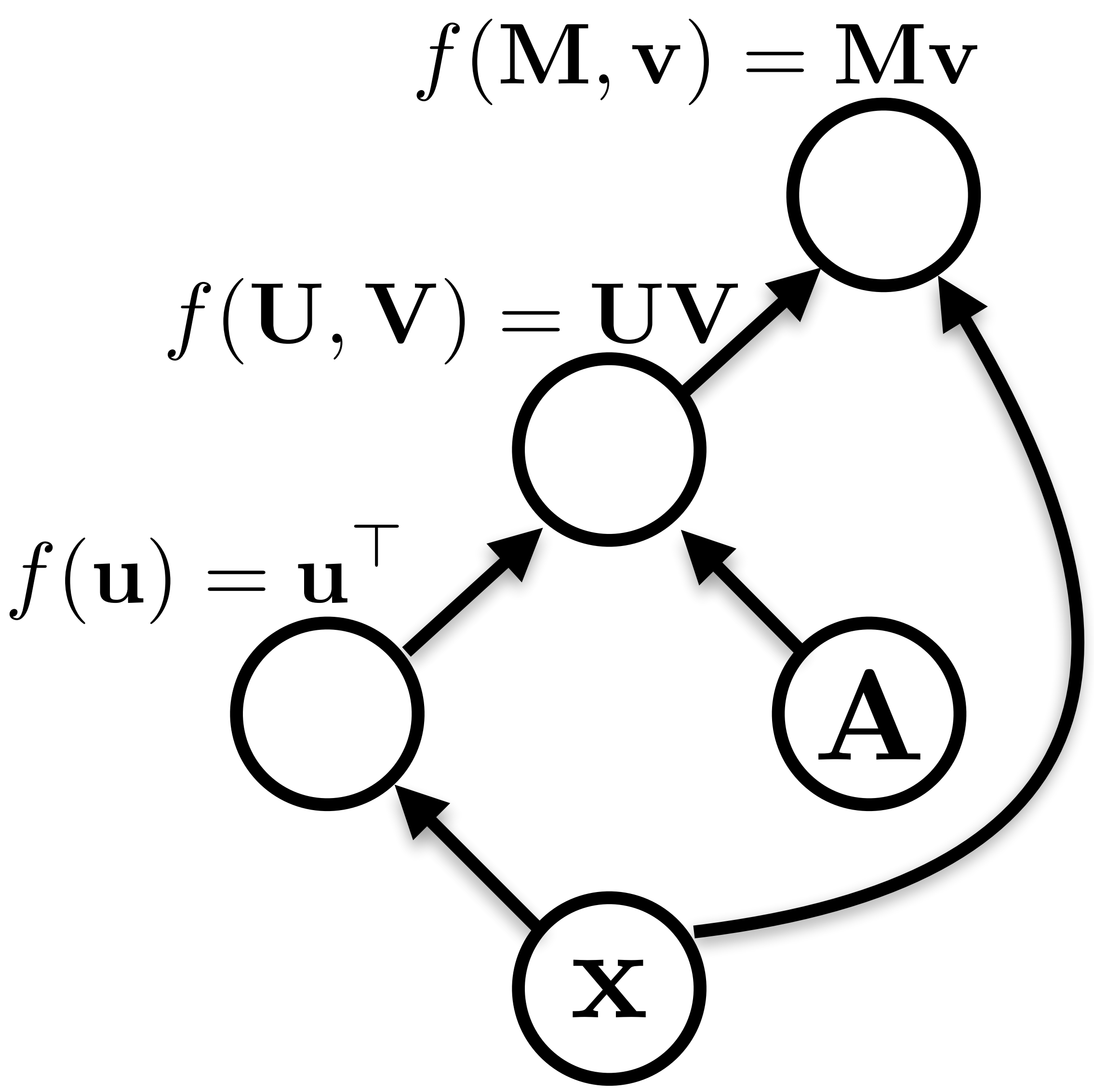


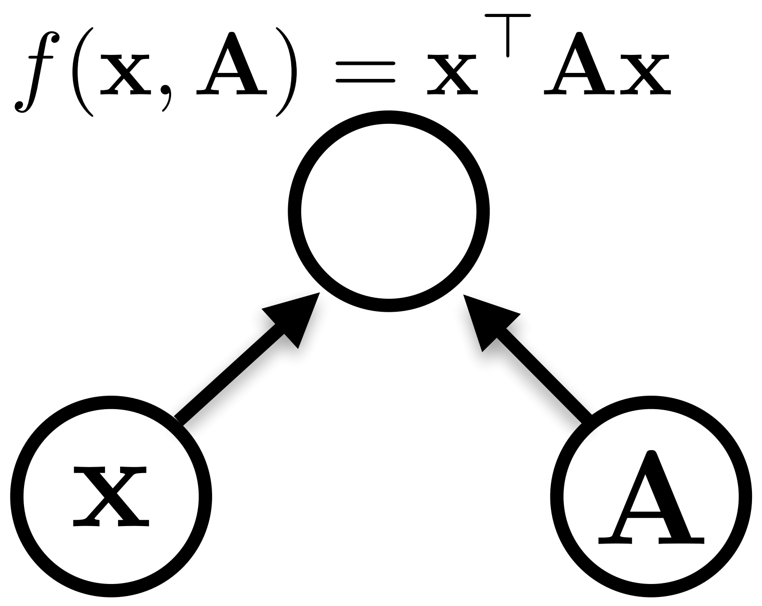

Computation graphs are generally directed and acyclic

expression:

$$\mathbf{x}^\top \mathbf{A}\mathbf{x}$$

graph:



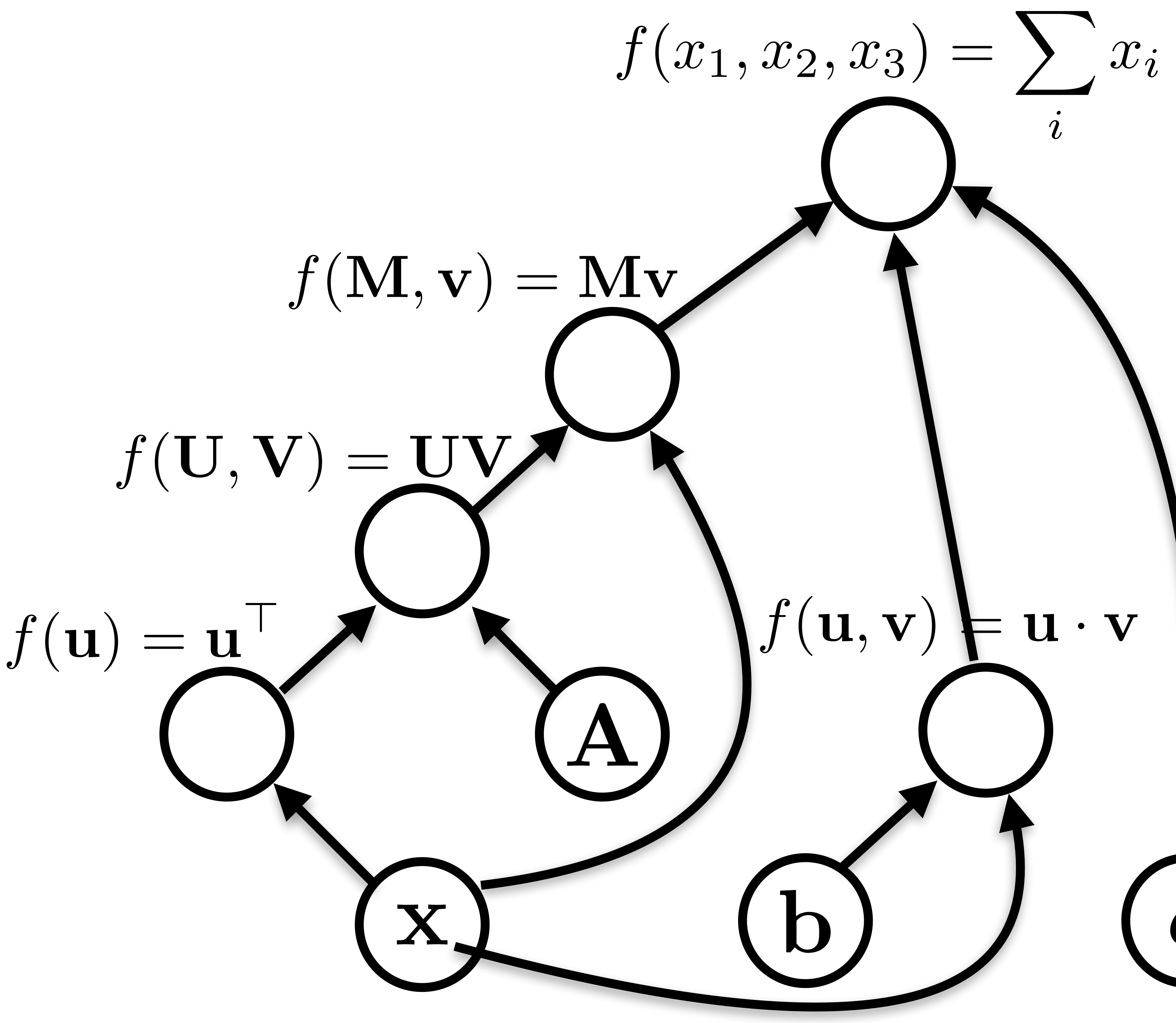

$$\frac{\partial f(\mathbf{x}, \mathbf{A})}{\partial \mathbf{x}} = (\mathbf{A}^\top + \mathbf{A})\mathbf{x}$$

$$\frac{\partial f(\mathbf{x}, \mathbf{A})}{\partial \mathbf{A}} = \mathbf{x}\mathbf{x}^\top$$

expression:

$$\mathbf{x}^\top \mathbf{A}\mathbf{x} + \mathbf{b} \cdot \mathbf{x} + c$$

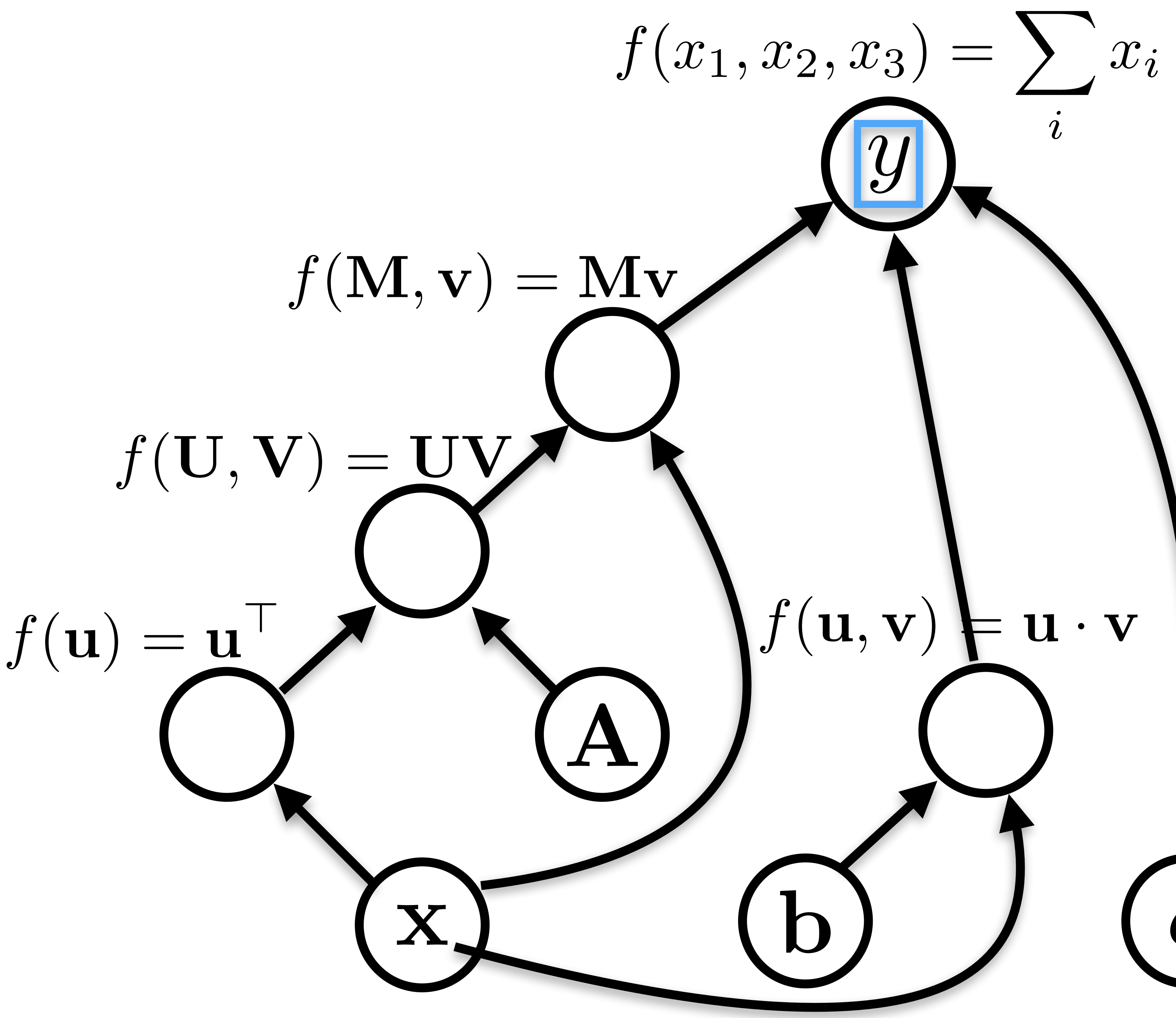
graph:



expression:

$$\boxed{y =} \mathbf{x}^\top \mathbf{A} \mathbf{x} + \mathbf{b} \cdot \mathbf{x} + c$$

graph:



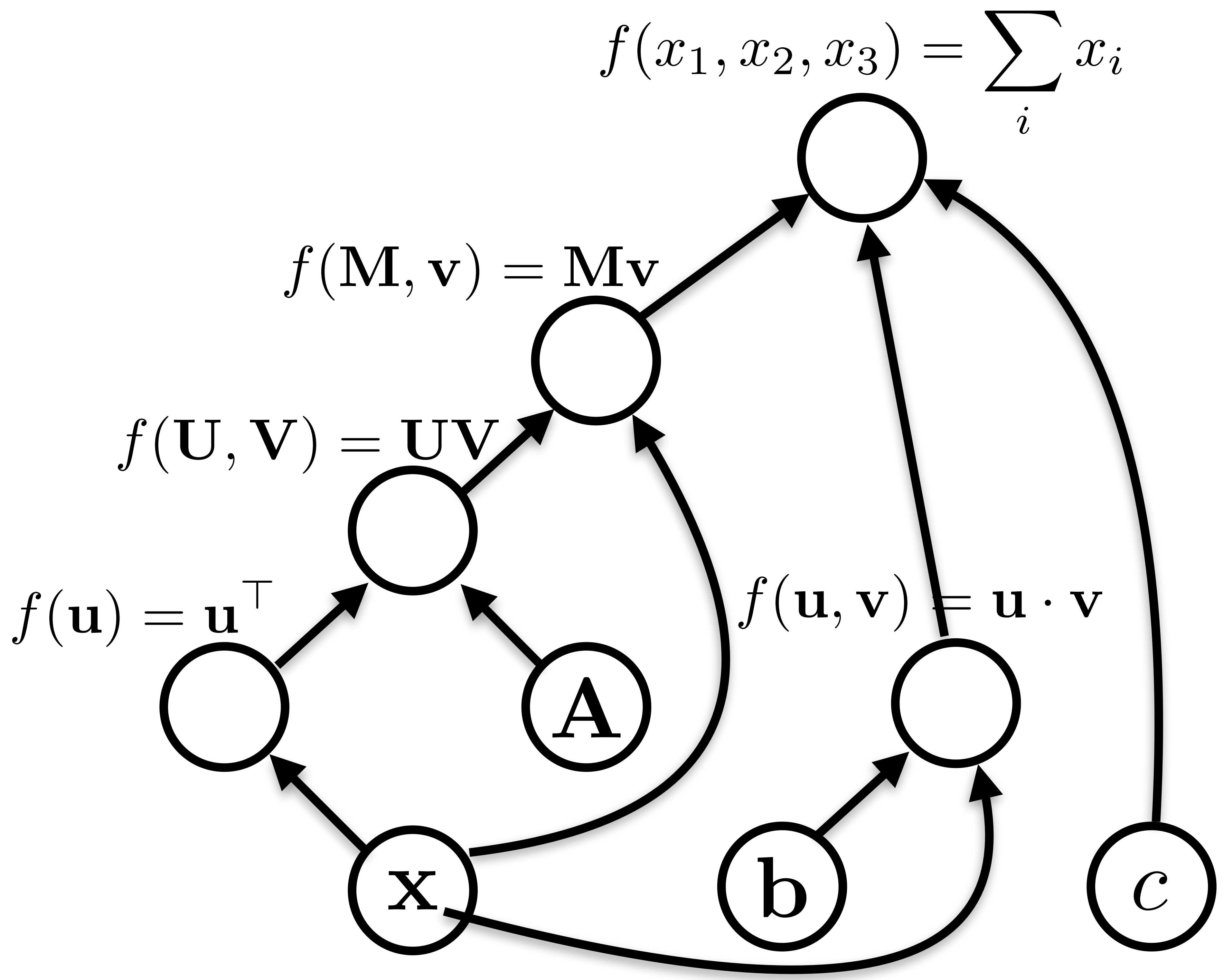

variable names are just labelings of nodes.

# Algorithms (1)

- Graph construction
- Forward propagation
  - In topological order, compute the value of the node given its inputs

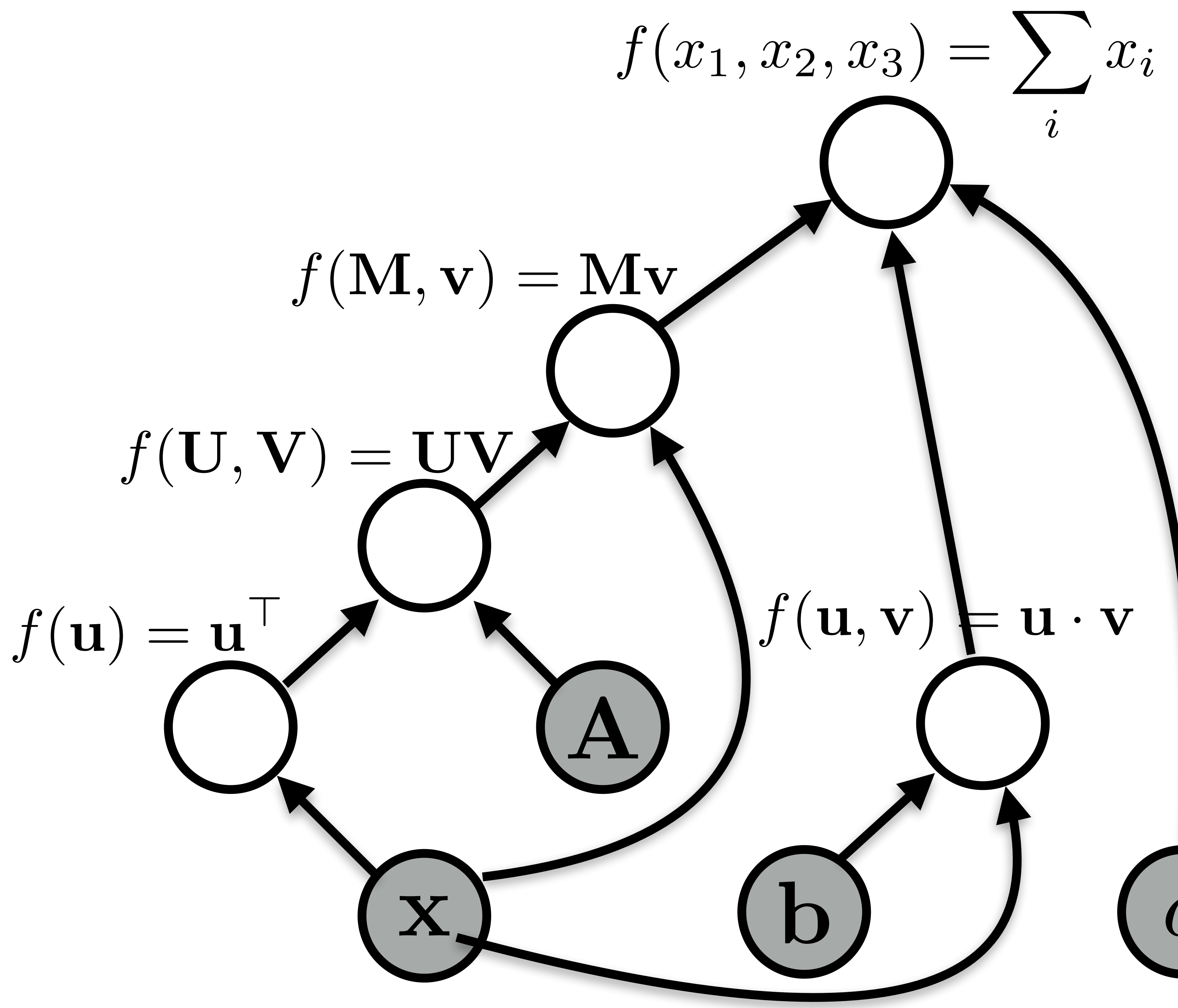
# Forward Propagation

graph:



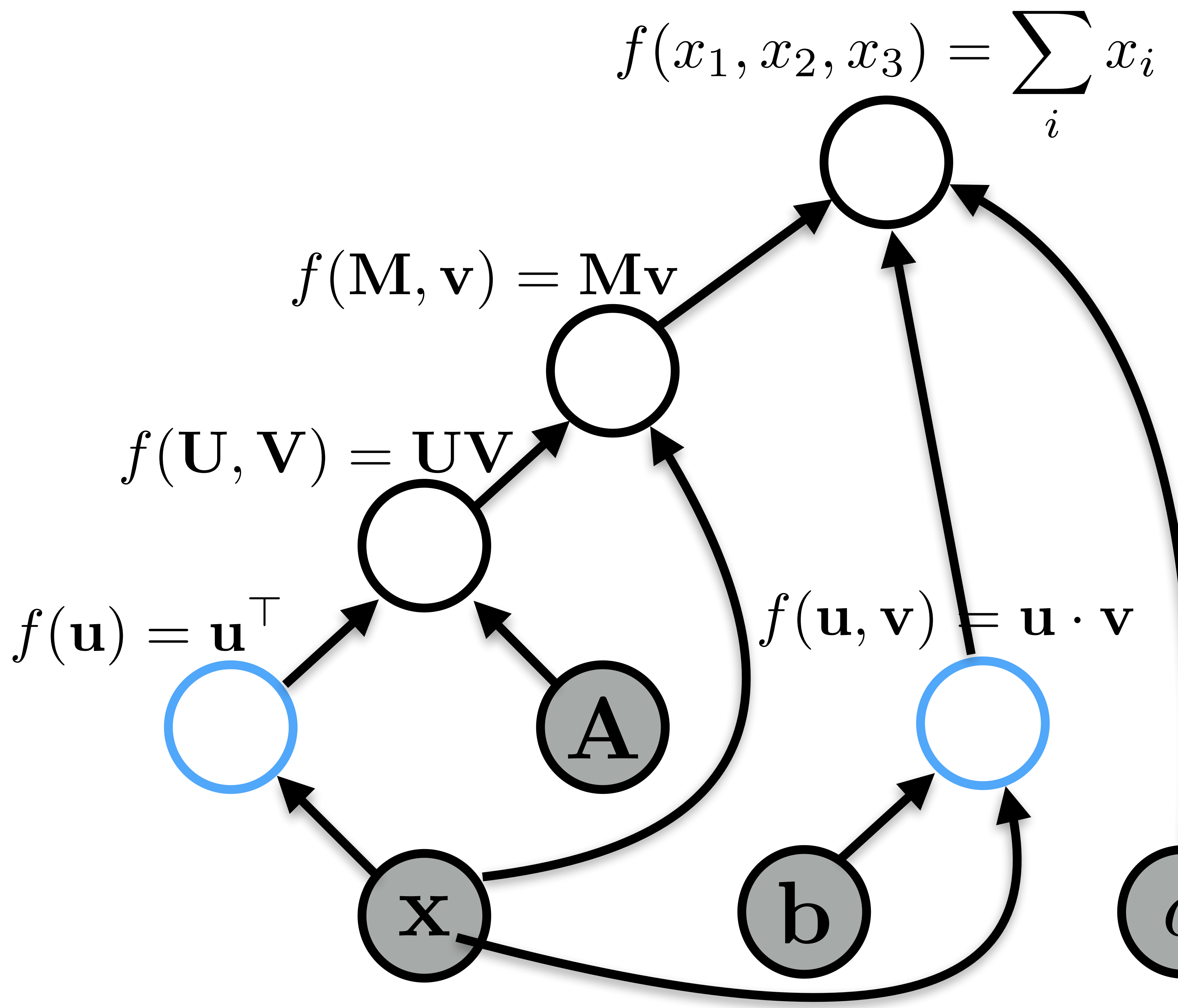
# Forward Propagation

graph:



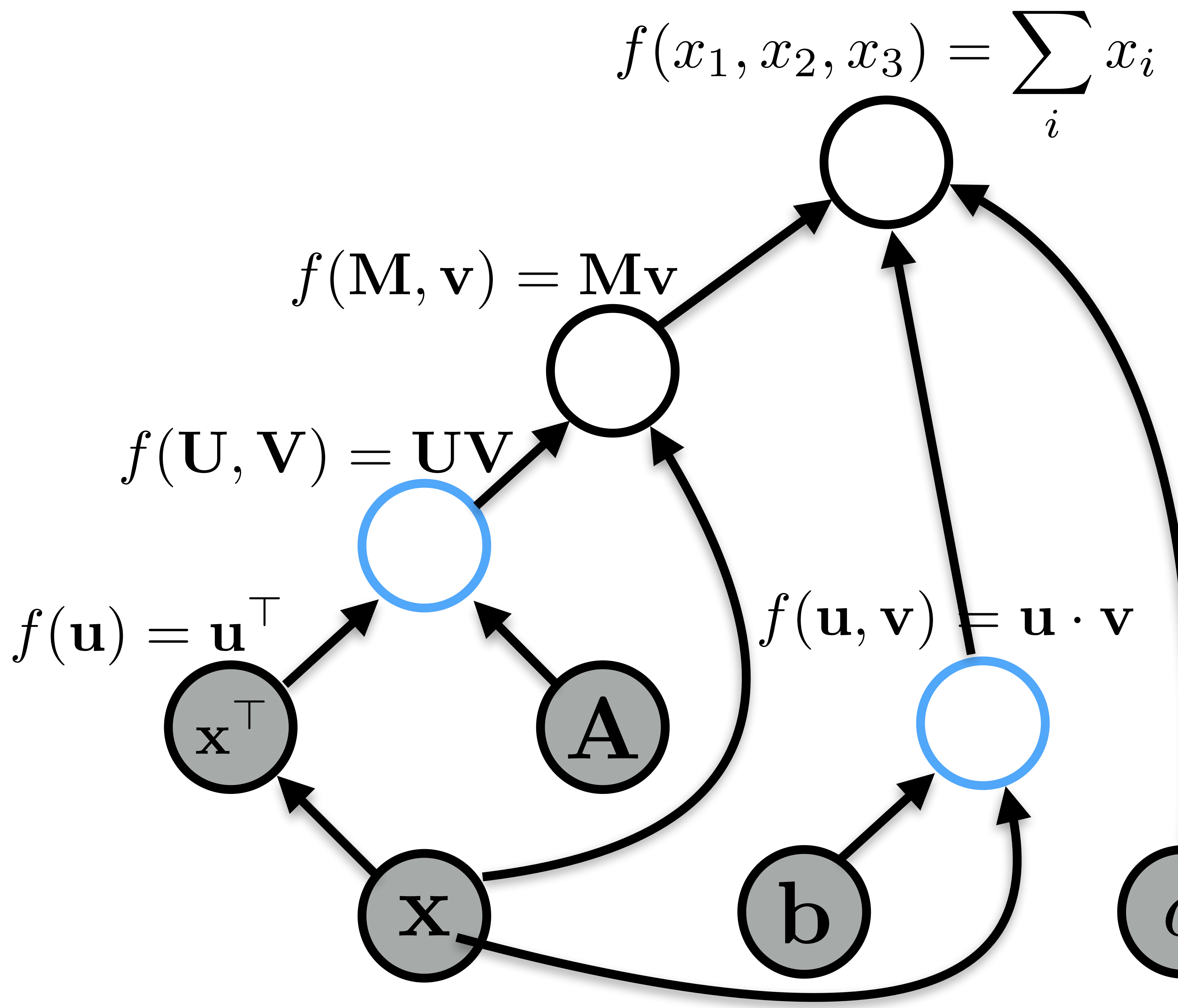
# Forward Propagation

graph:



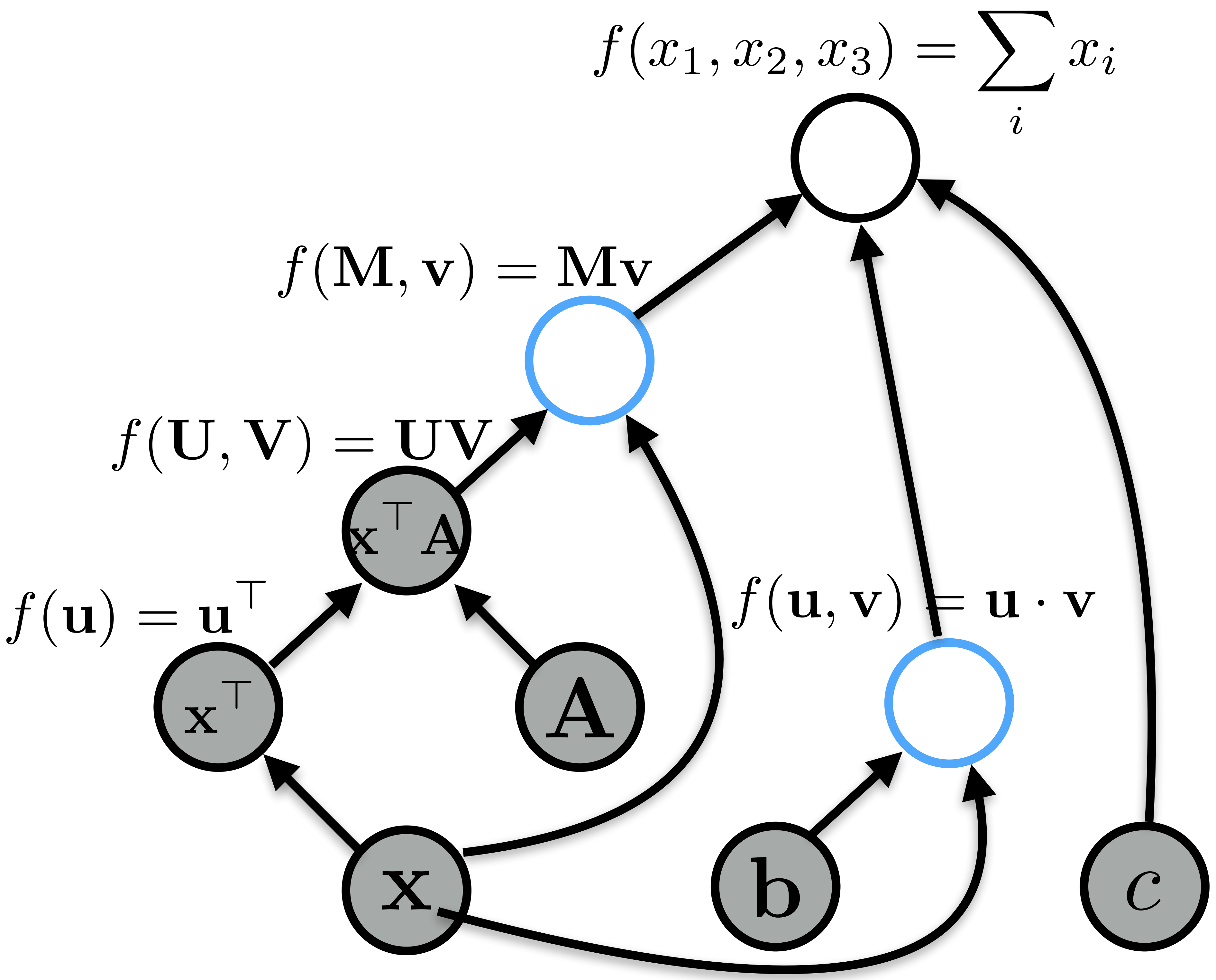
# Forward Propagation

graph:



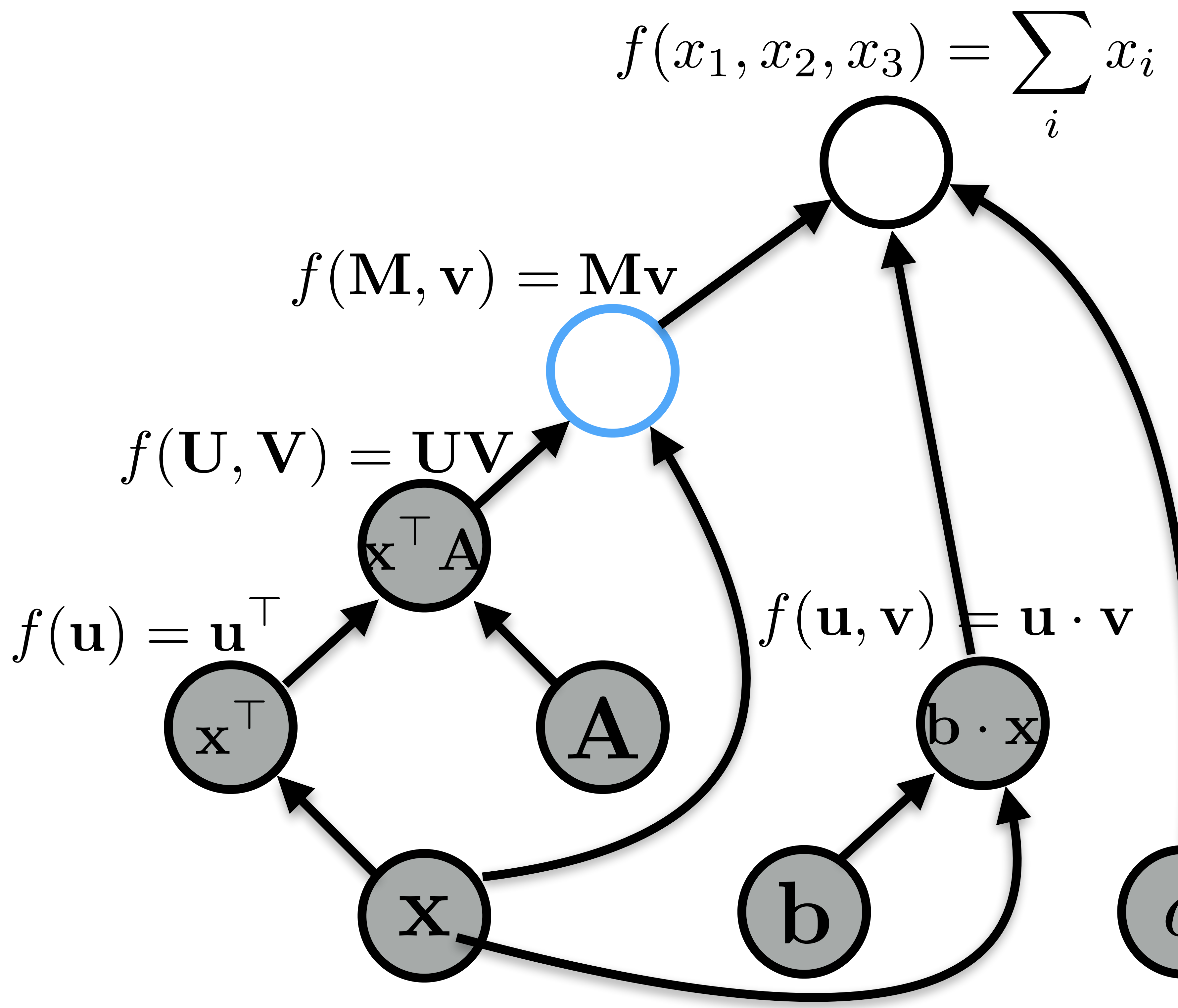
# Forward Propagation

graph:



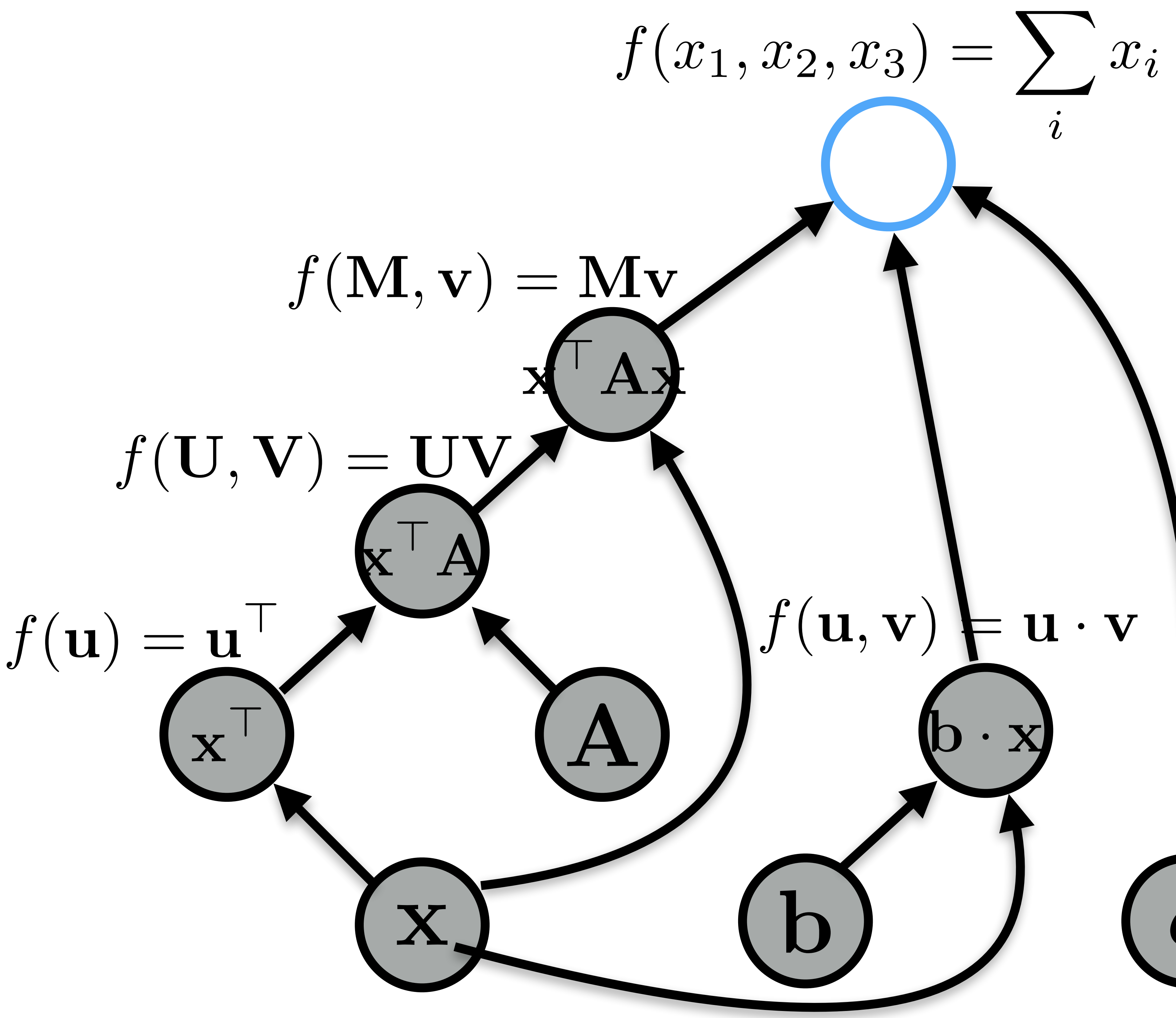
# Forward Propagation

graph:



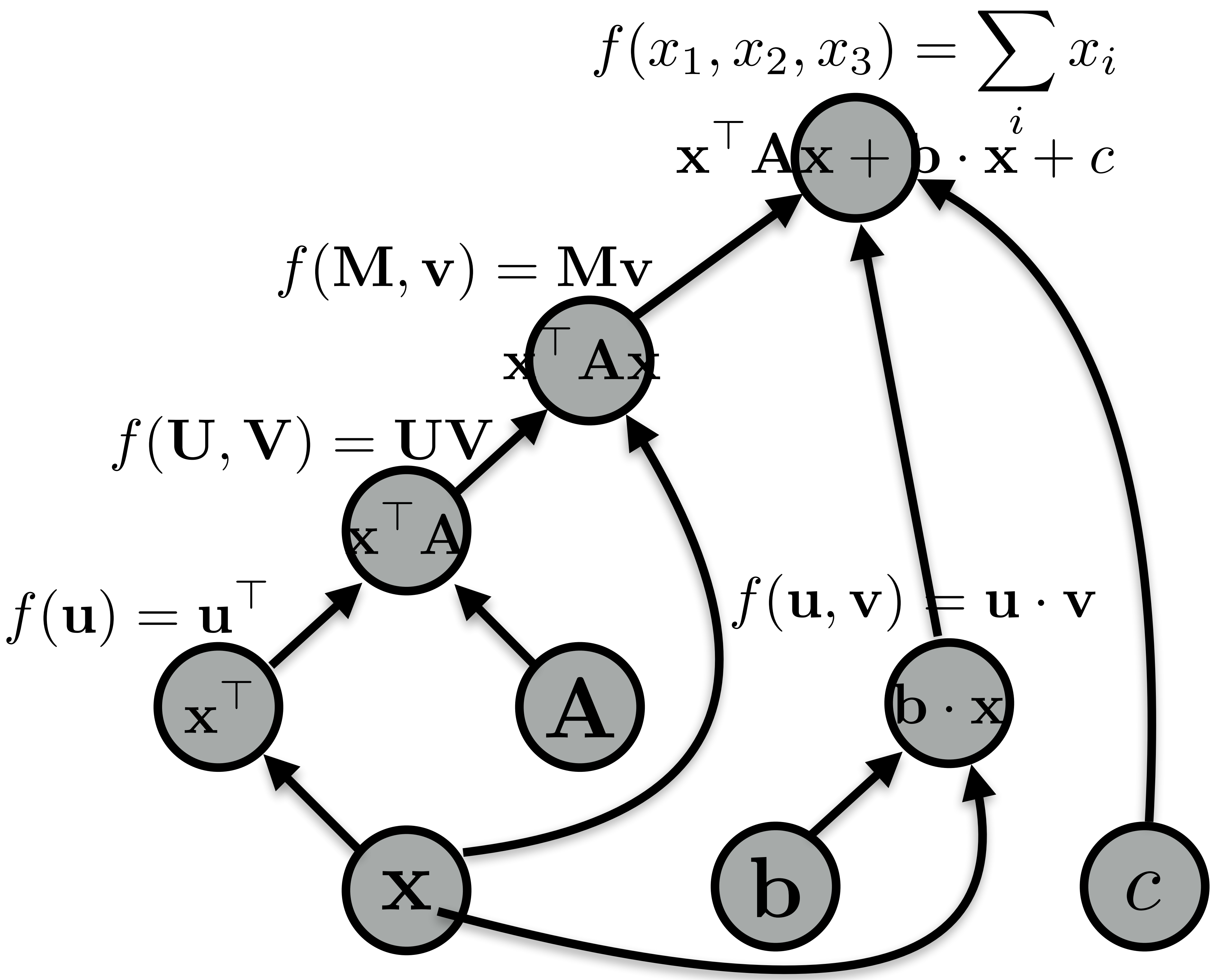
# Forward Propagation

graph:



# Forward Propagation

graph:

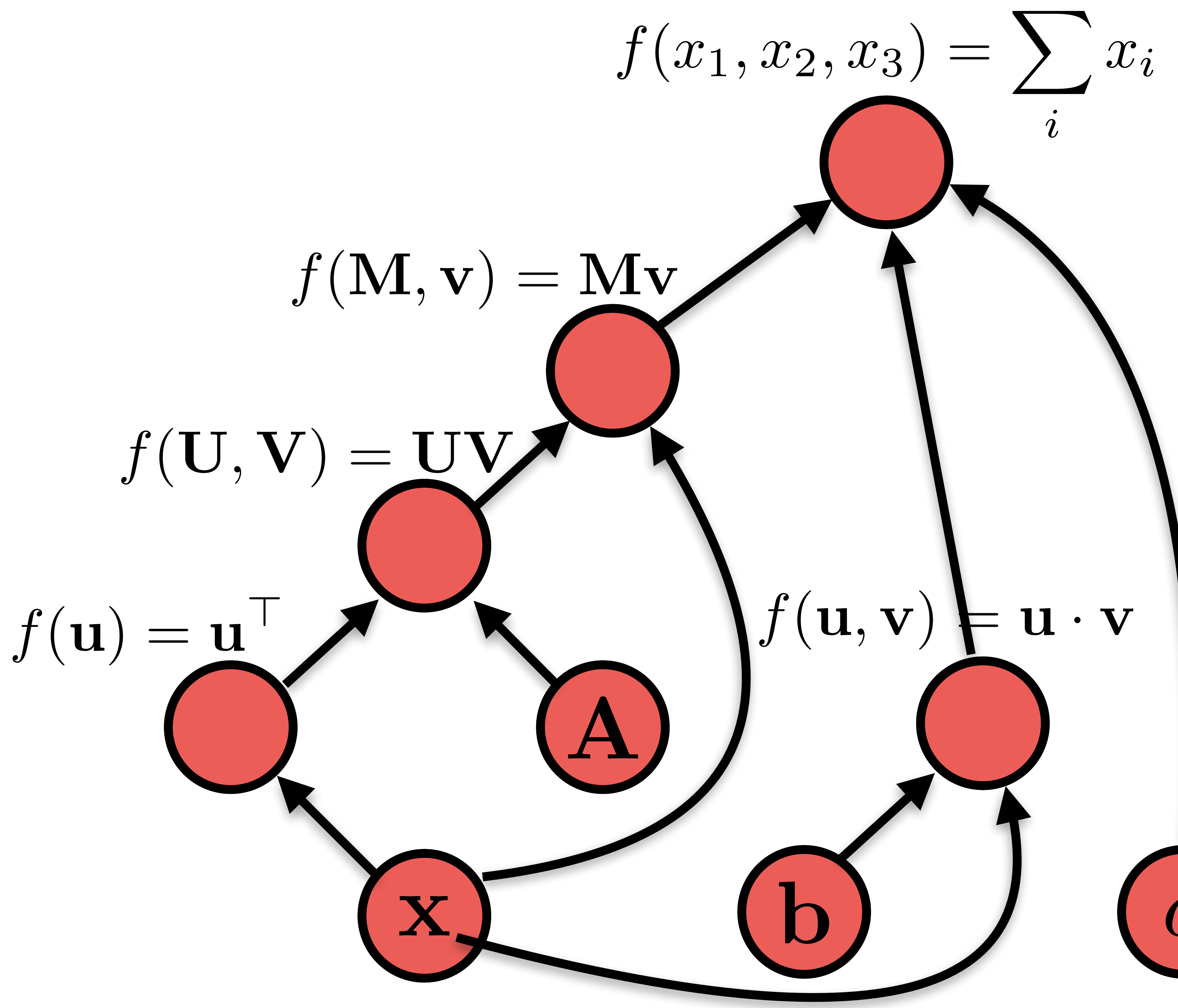


# Algorithms (2)

- Back-propagation:
  - Process examples in reverse topological order
  - Calculate the derivatives of the parameters with respect to the final value
    (This is usually a “loss function”, a value we want to minimize)
- Parameter update:
  - Move the parameters in the direction of this derivative
    W -= α * dl/dW

# Back Propagation

graph:



# Neural Network Frameworks

PYTORCH

Examples in this class

TensorFlow

**JAX**

Autograd and XLA
with numpy

# Basic Process in (Dynamic) Neural Network Frameworks

- Create a model
- For each example
  - **create a graph** that represents the computation you want
  - **calculate the result** of that computation
  - if training, perform **back propagation and update**

# Pytorch Quick Tutorial

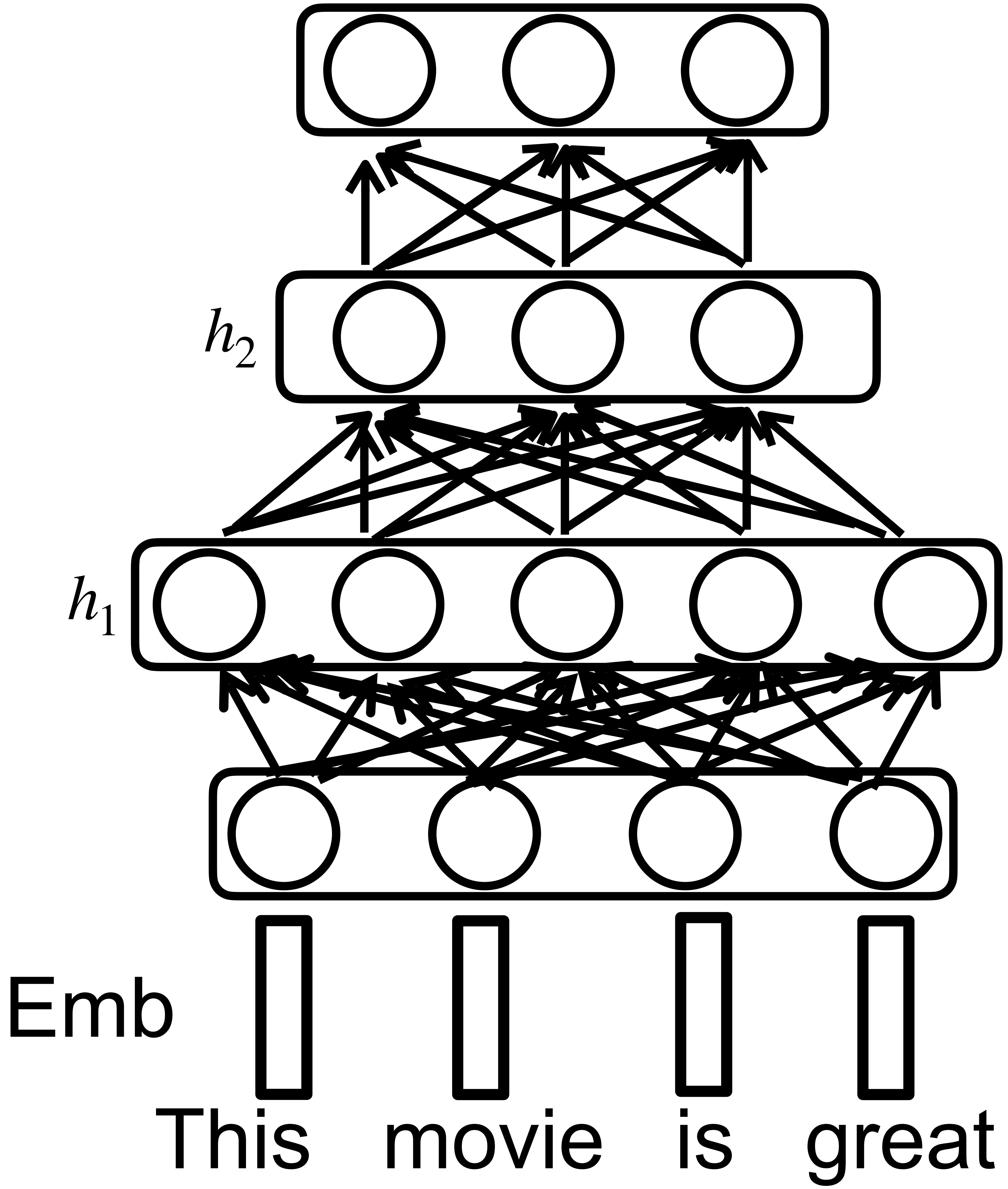
https://pytorch.org/tutorials/beginner/basics/intro.html

# Feedforward Neural Net (FFN)

- also known as multilayer perceptron (MLP)
- Layers are connected sequentially
- Each layer has full-connection (each unit is connected to all units of next layer)
  - Linear project followed by
  - an element-wise nonlinear activation function

$$h = \sigma(w \cdot x + b)$$

- There is no connection from output to input



# Recurrent Neural Networks

# Long-distance Dependencies in Language

- Agreement in number, gender, etc.

**He** does not have very much confidence in **himself**.

**She** does not have very much confidence in **herself**.

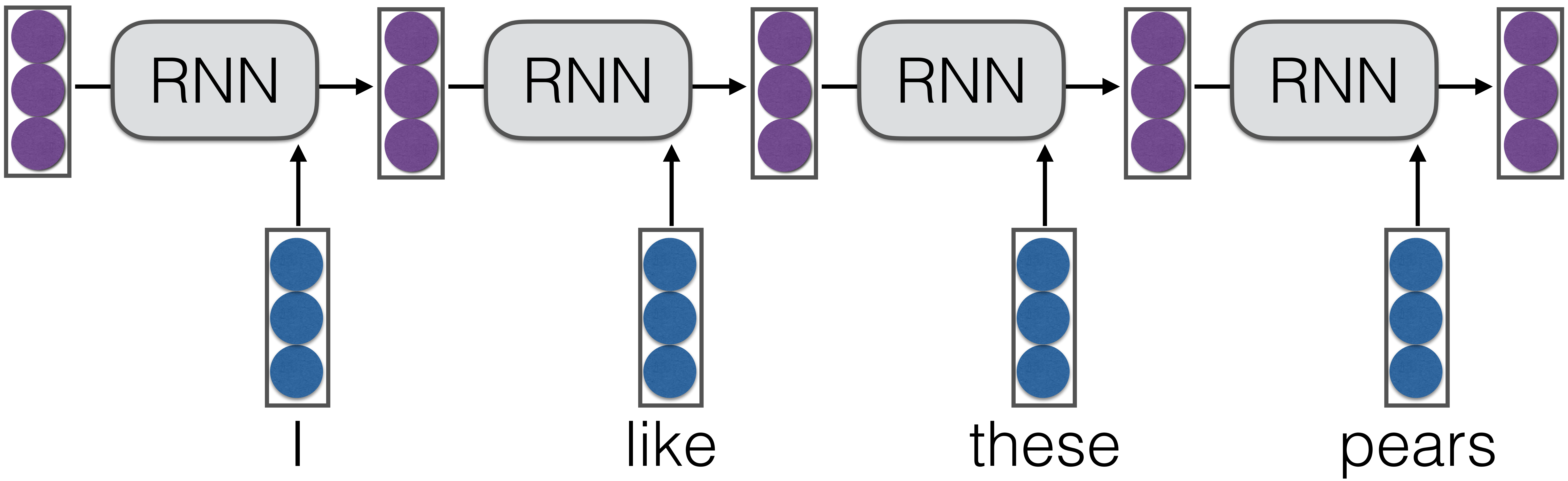
- Selectional preference

The **reign** has lasted as long as the life of the **queen**.

The **rain** has lasted as long as the life of the **clouds**.

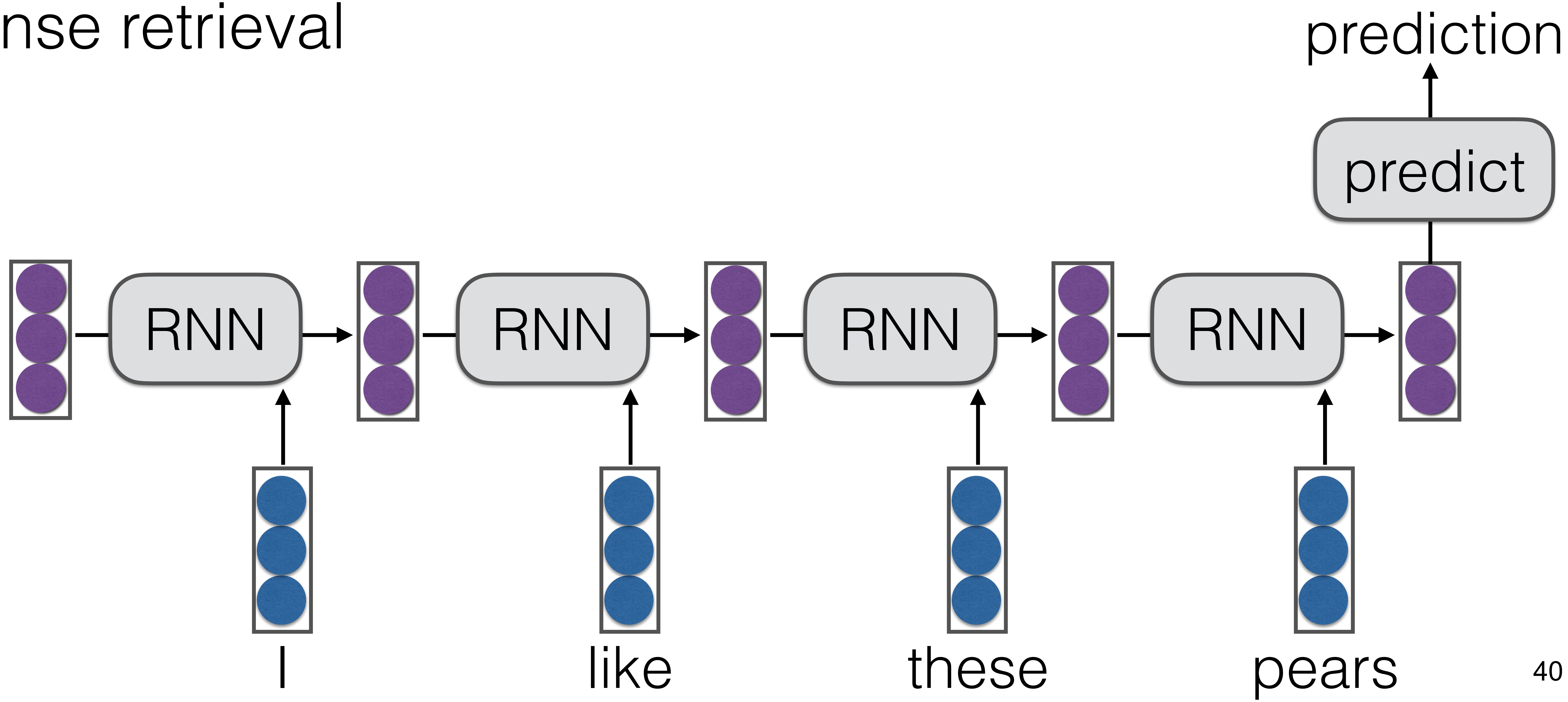
# Recurrent Neural Networks (Elman 1990)

- Tools to “remember” information



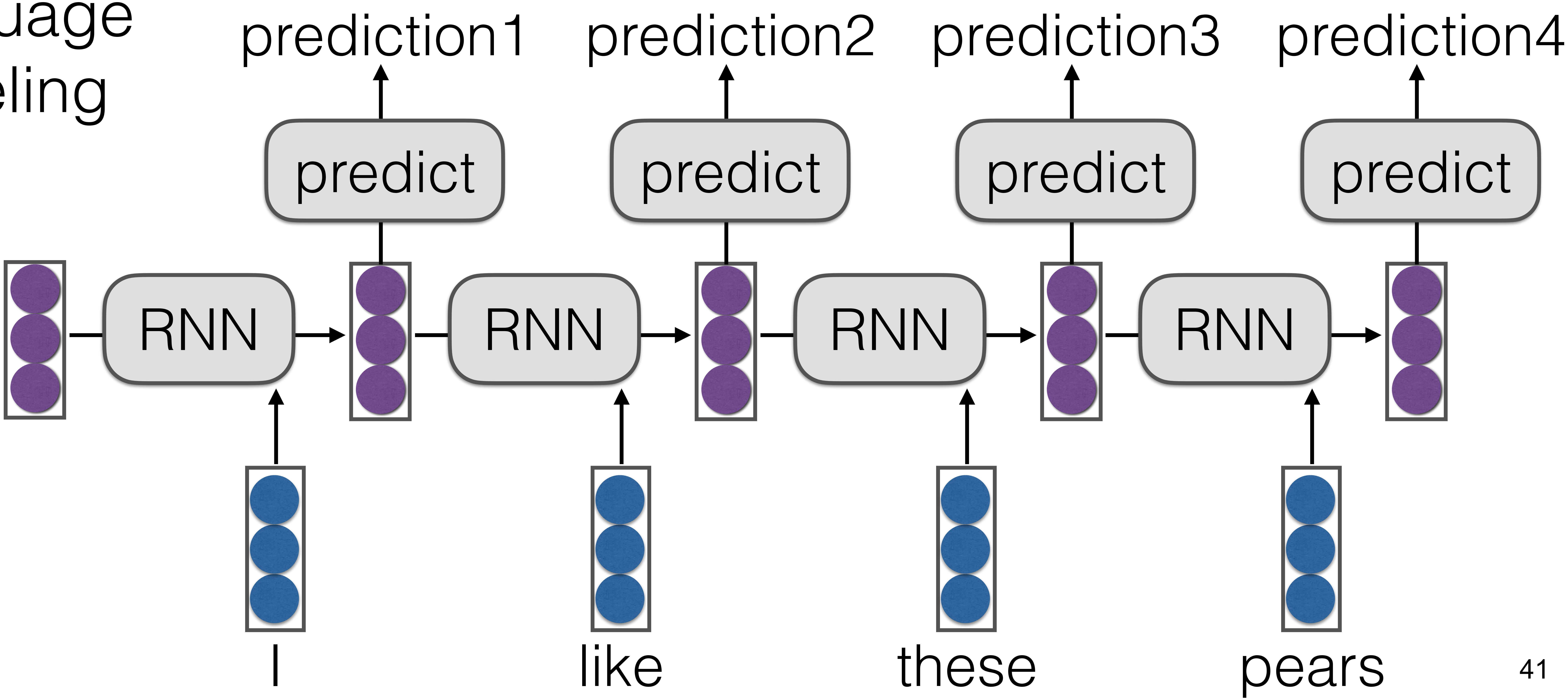
# Sentence Representation for Downstream Tasks

- Text classification
- Conditional generation
- Sentense retrieval



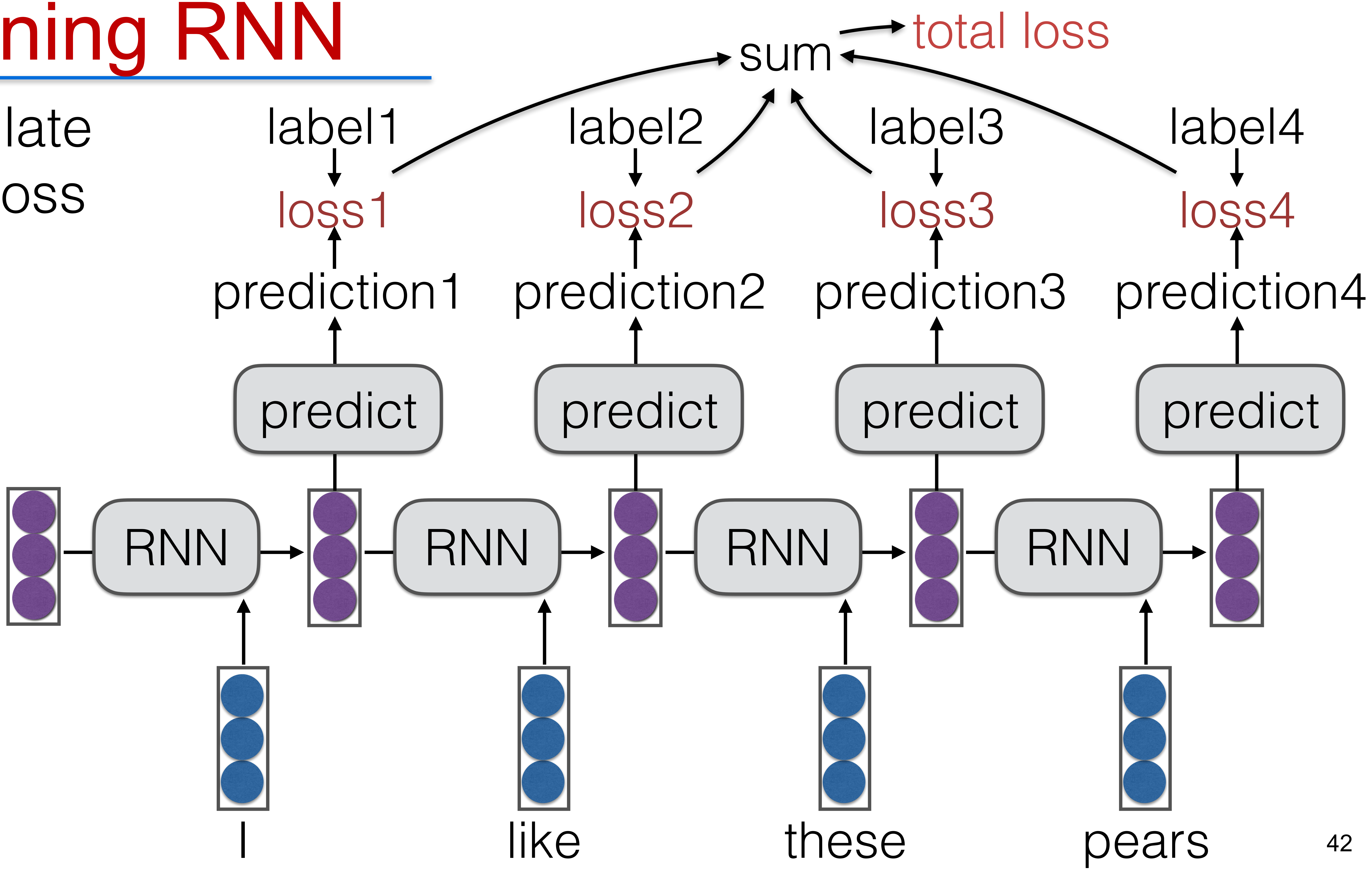
# Representing Words

- Sequence Labeling
- Language Modeling



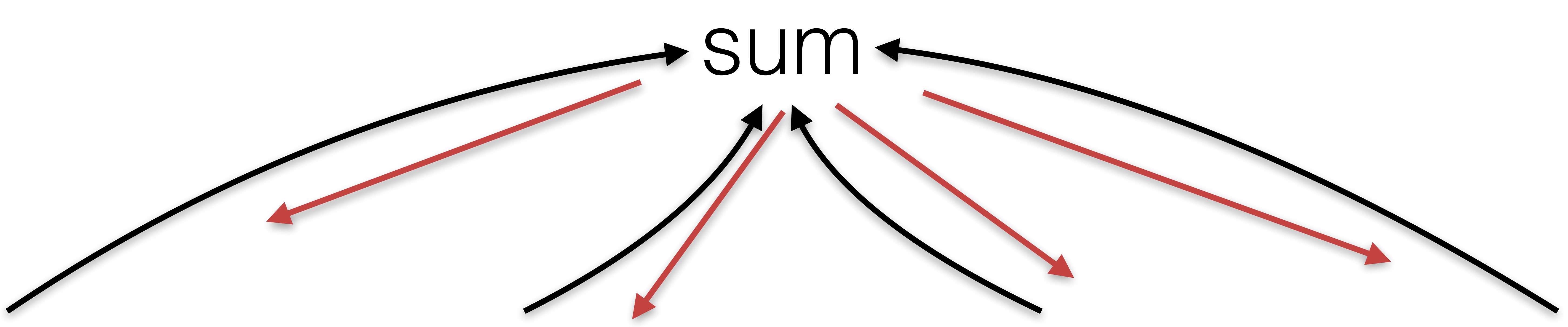
# Training RNN

- calculate total loss



# RNN Training

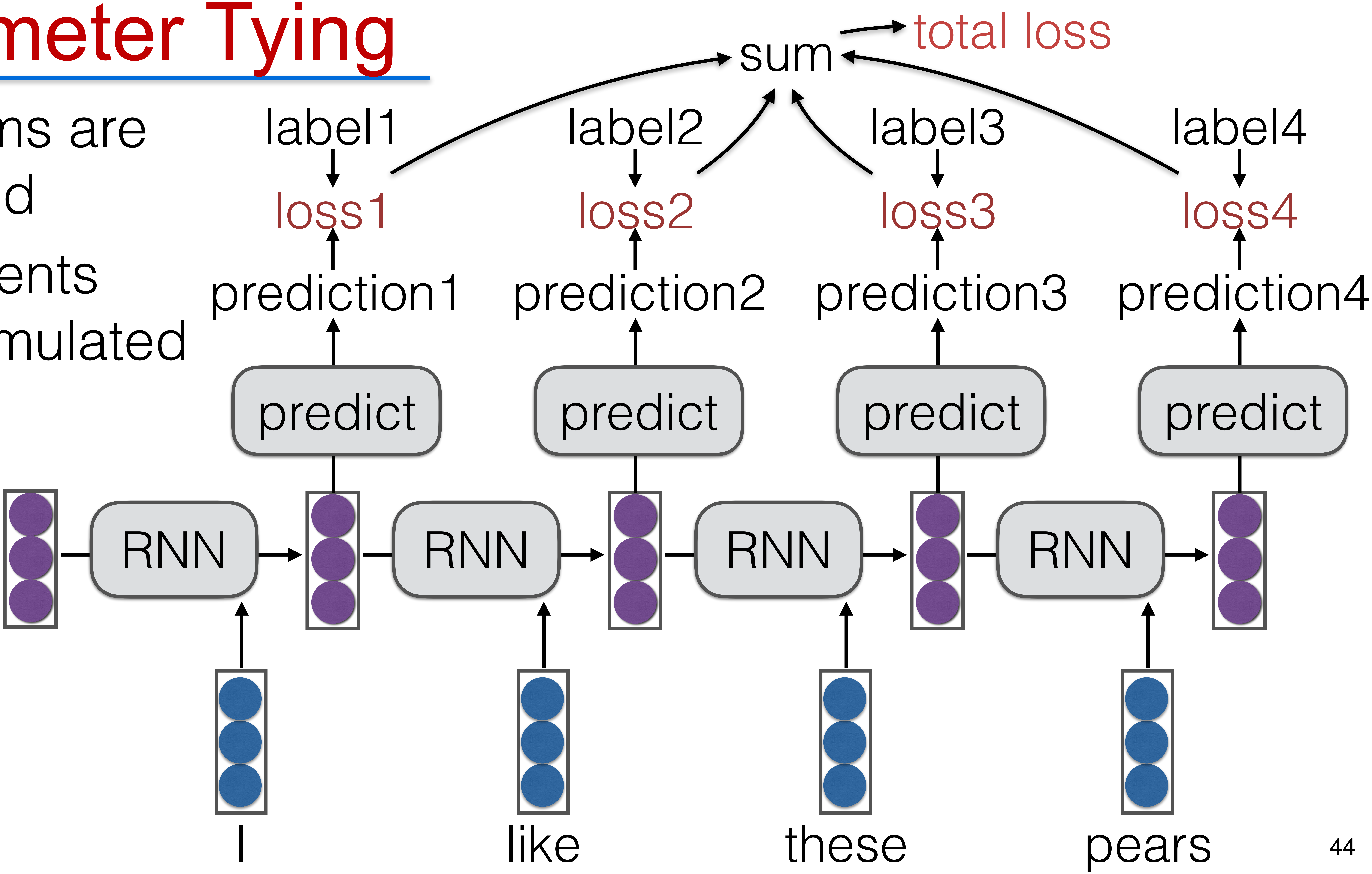
- The unrolled graph is a well-formed (DAG) computation graph—we can run backprop



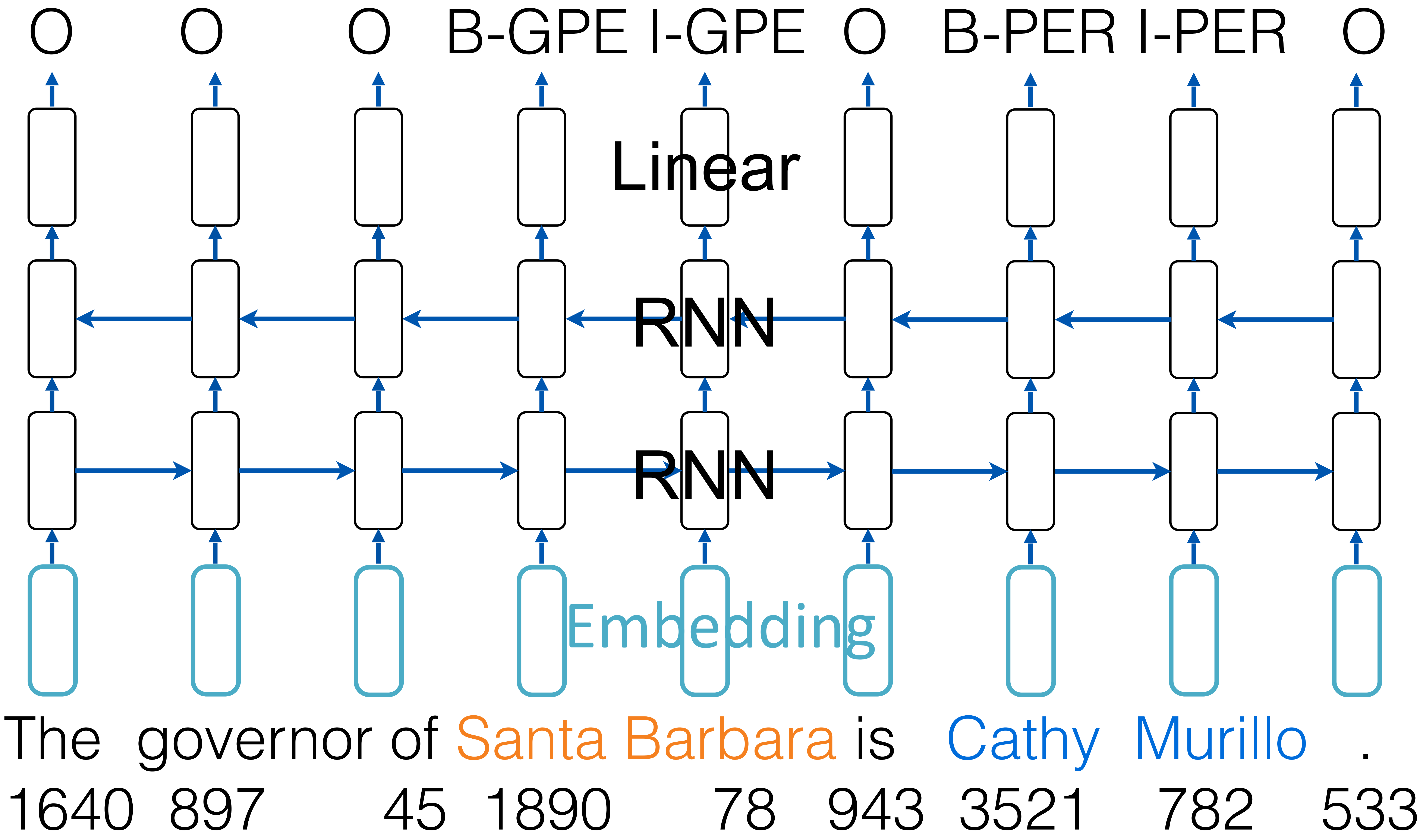

  - Parameters are tied across time, derivatives are aggregated across all time steps
  - This is historically called “backpropagation through time” (BPTT)

# Parameter Tying

- Params are shared
- gradients accumulated



# Bi-directional RNN



# Multilingual Labeling/Classification Data and Models

# Language Identification

*LTI Language Identification Corpus*

http://www.cs.cmu.edu/~ralf/langid.html

Benchmark on 1152 languages from a variety of free sources

*langid.py*

https://github.com/saffsd/langid.py

Off-the-shelf language ID system for 90+ languages

*Automatic Language Identification in Texts: A Survey*

https://arxiv.org/pdf/1804.08186.pdf

# Text Classification

- Very broad field, many different datasets

*MLDoc: A Corpus for Multilingual Document Classification in Eight Languages*
https://github.com/facebookresearch/MLDoc
Topic classification, eight languages

*PAWS-X: Paraphrase Adversaries from*
https://github.com/google-research-datasets/paws/tree/
Paraphrase detection (sentence *pair* classification)

*Cross-lingual Natural Language Inference (XNLI) corpus*
https://cims.nyu.edu/~sbowman/
Textual entailment prediction (sentence *pair* classification)

*Cross-lingual Sentiment Classification*
Available from: https://github.com/ccsasuke/
Chinese-English cross-lingual sentiment dataset

# Part of Speech/Morphological Tagging

- Part of universal dependencies treebank
  https://universaldependencies.org/
- Contains parts of speech and morphological features for 90 languages
- Standardized "Universal POS" and "Universal Morphology" tag sets make things consistent
- Several pre-trained models on these datasets:
  - *Udify:* https://github.com/Hyperparticle/udify
  - *Stanza:* https://stanfordnlp.github.io/stanza/

# Named Entity Recognition

- “Gold standard” data
  - CoNLL 2002/2003 Language Independent Named Entity Recognition
  - https://www.clips.uantwerpen.be/conll2003/ner/
  - English, German, Spanish, Dutch human annotated data
- “Silver Standard”
  - WikiAnn Entity Recognition/Linking in 282 Languages
  - https://www.aclweb.org/anthology/P17-1178/
  - Available from: https://github.com/google-research/xtreme
  - Data automatically extracted from Wikipedia using inter-page links

# Composite Benchmarks

- Benchmarks that aggregate many different sequence labeling/classification tasks
- XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalization
  - 10 different tasks, 40 different languages
  - https://github.com/google-research/xtreme
- XGLUE: A New Benchmark Dataset for Cross-lingual Pre-training, Understanding and Generation
  - https://microsoft.github.io/XGLUE/
  - 11 tasks over 19 languages (including generation)

# Discussion Today

Assignment 1 introduction

Code walk