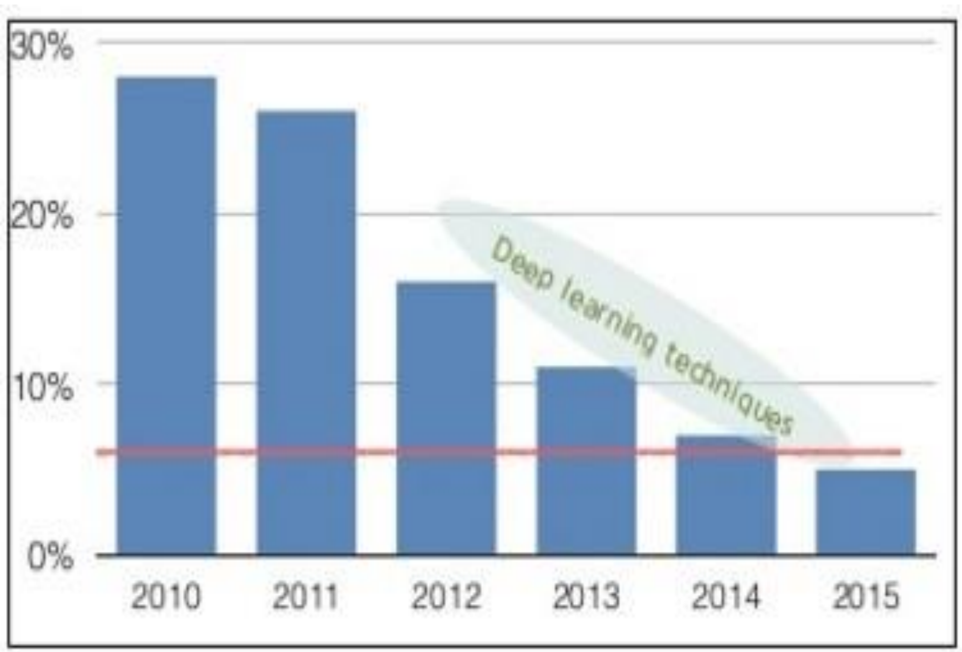
Deep Learning for Recommender Systems

Alexandros Karatzoglou (Scientific Director @ Telefonica Research)
alexk@tid.es, @alexk_z

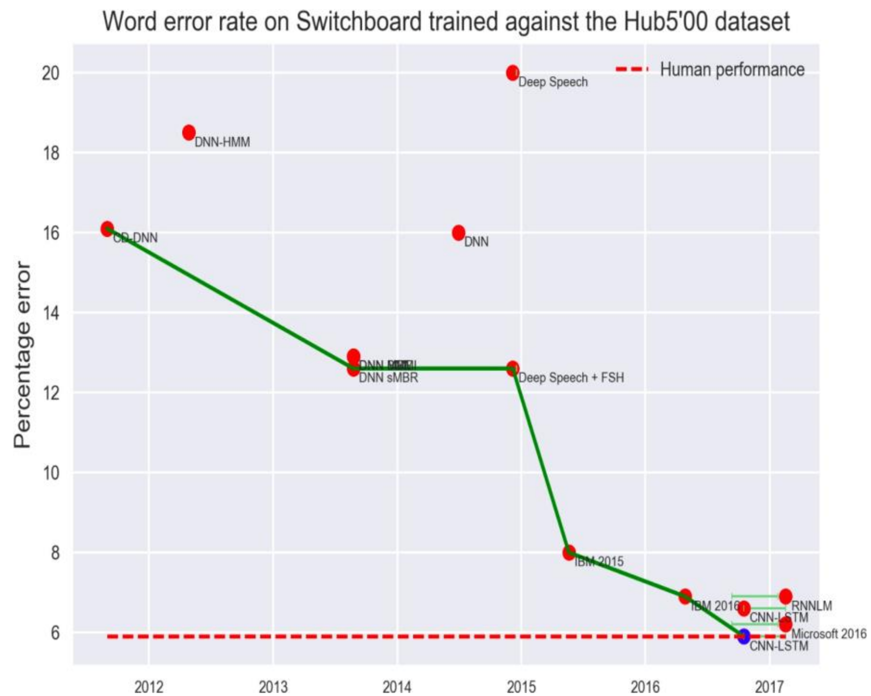
Balázs Hidasi (Head of Research @ Gravity R&D)
balazs.hidasi@gravityrd.com, @balazshidasi

RecSys’17, 29 August 2017, Como

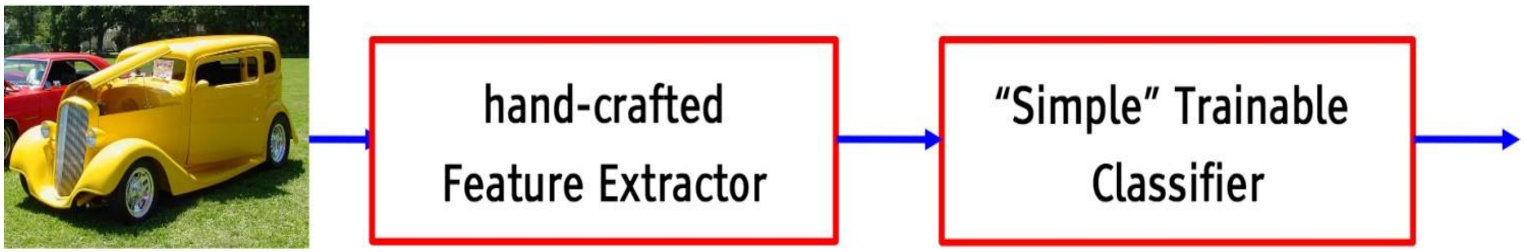
Why Deep Learning?

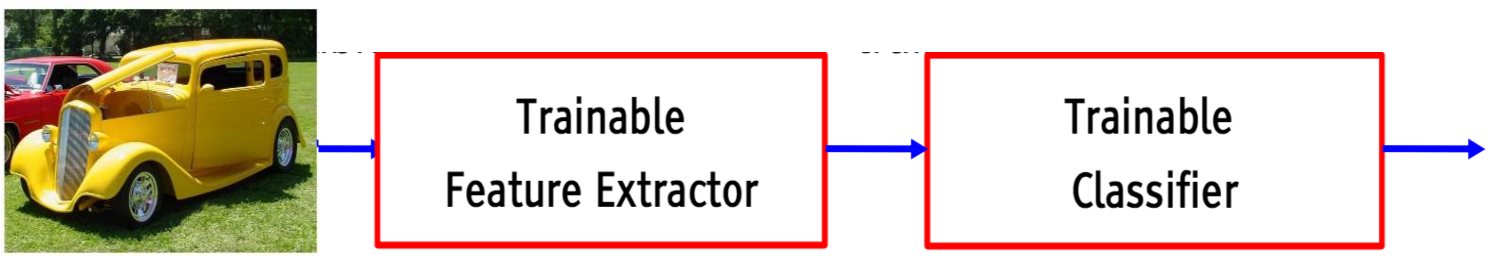


ImageNet challenge error rates (red line = human performance)

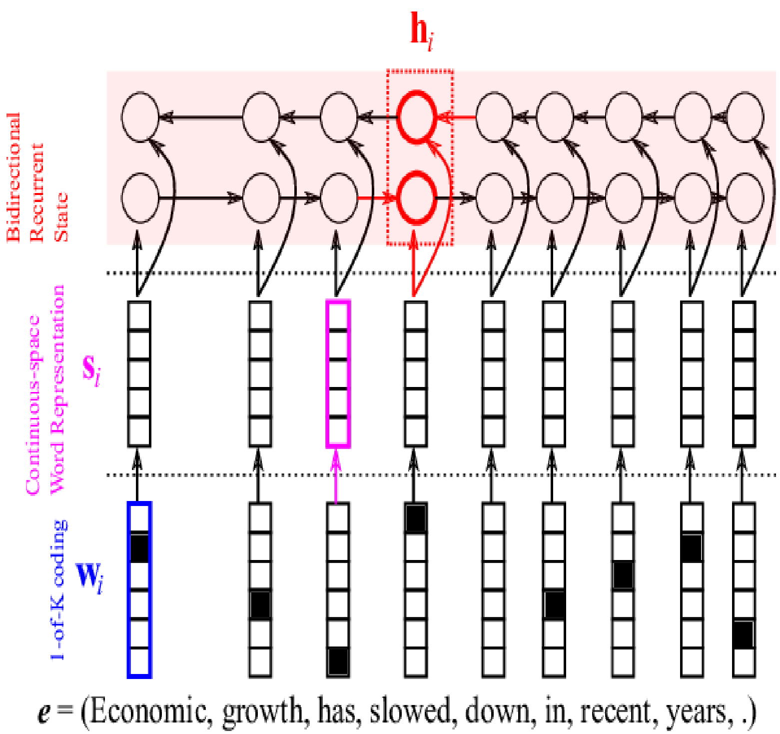


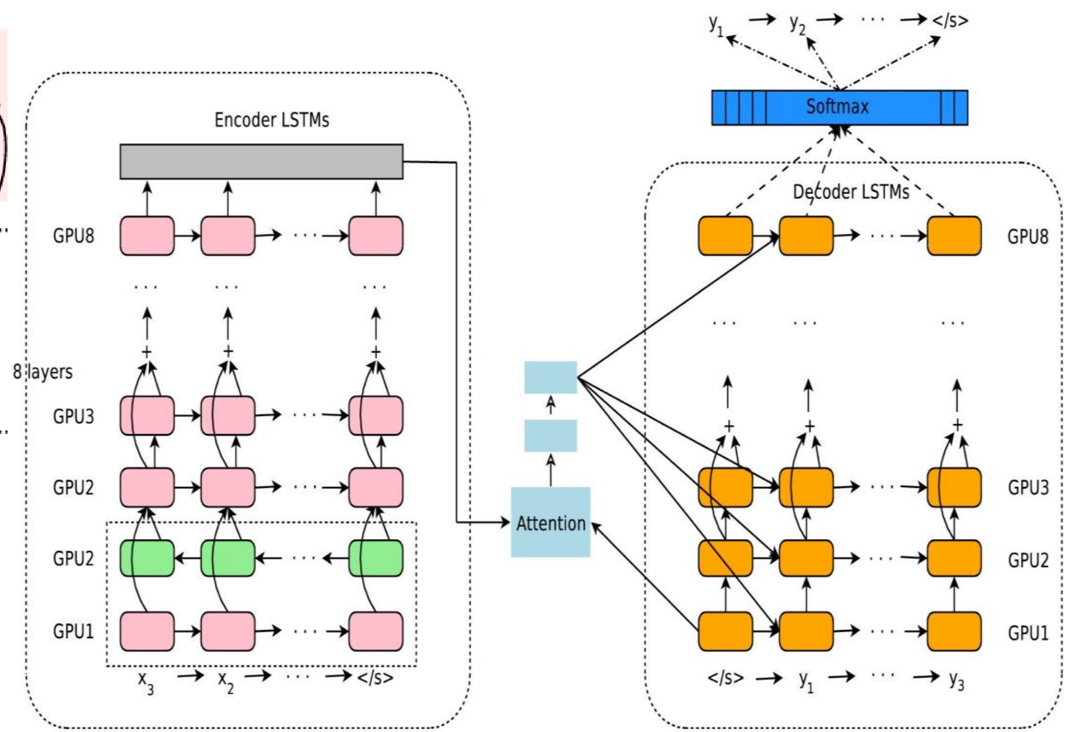
Why Deep Learning?



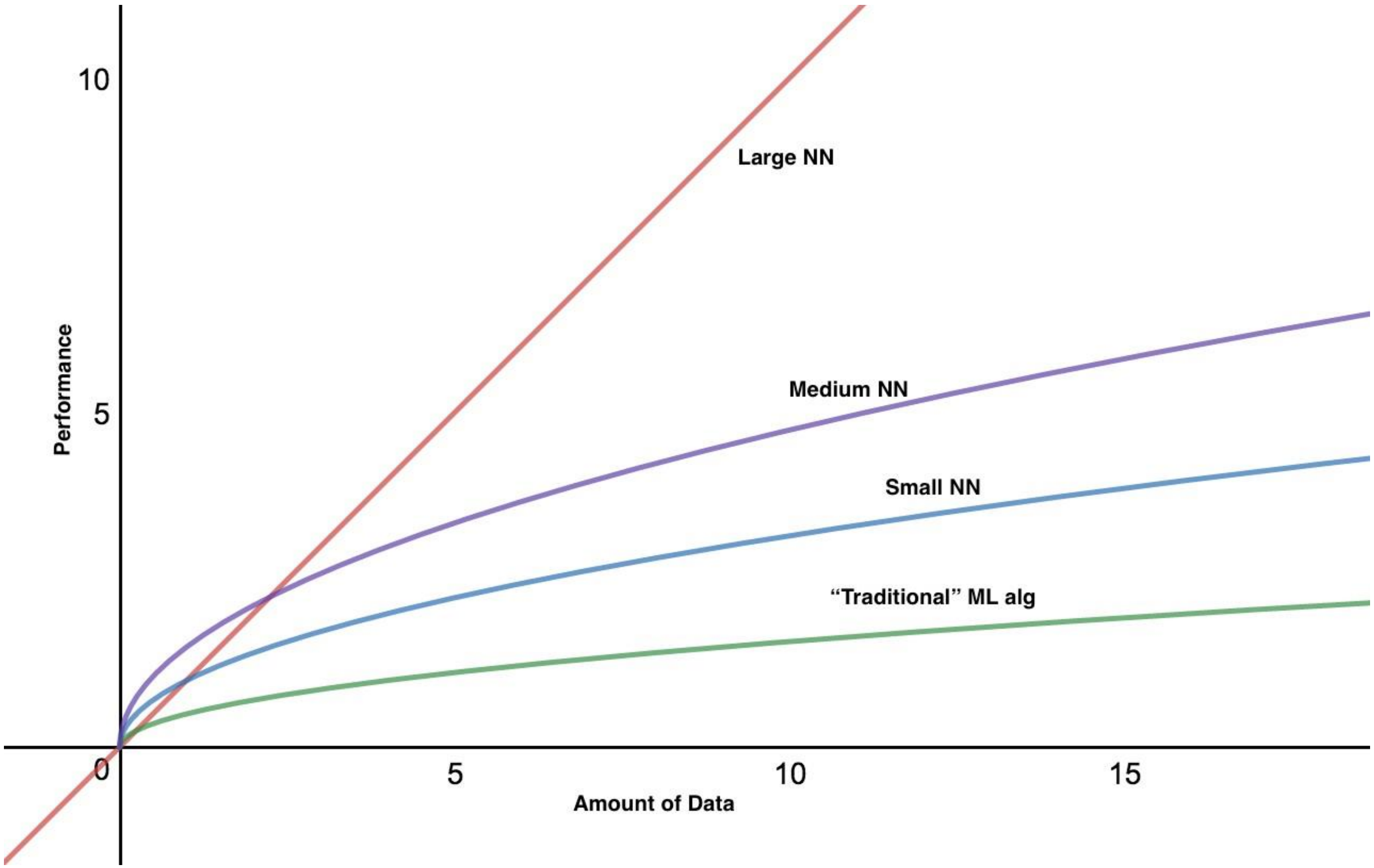


Complex Architectures



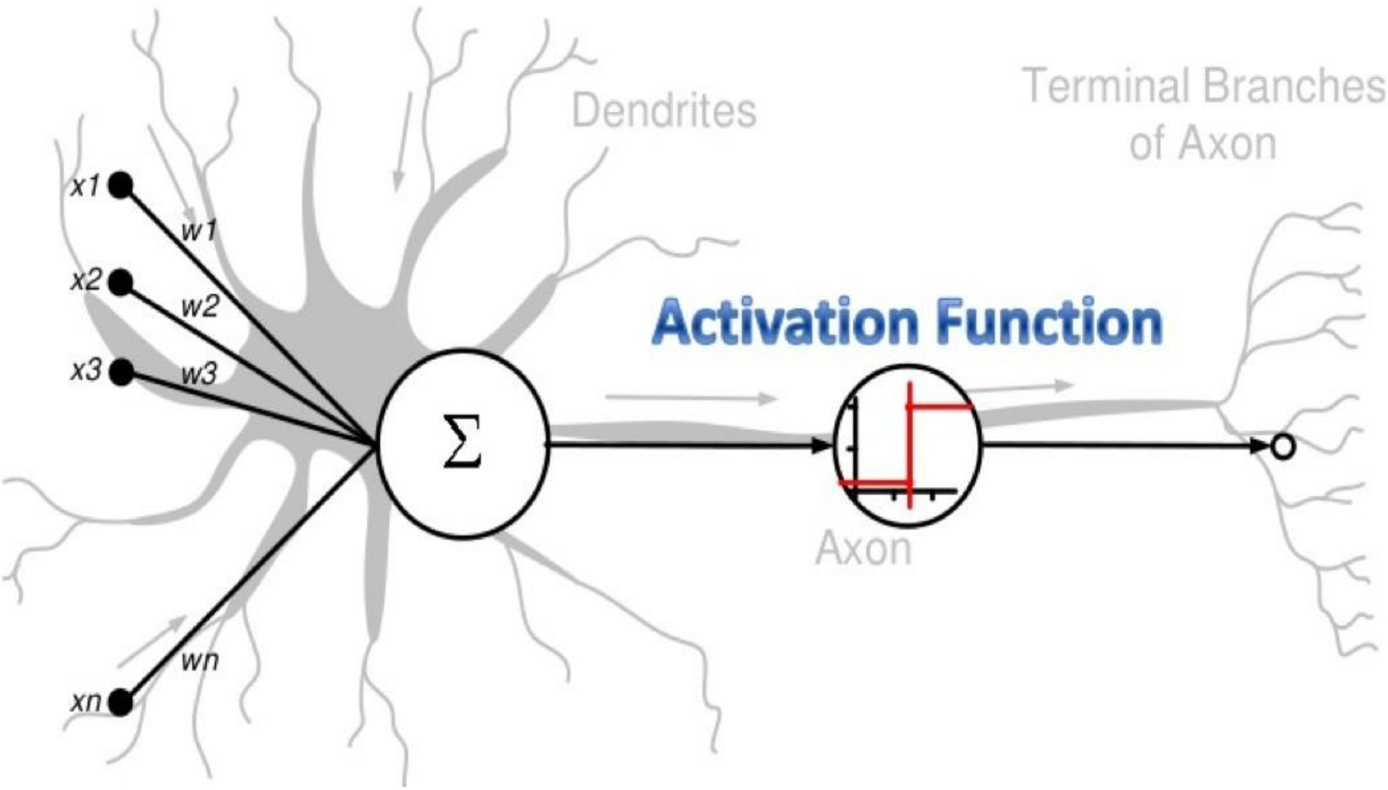


Neural Networks are Universal Function Approximators

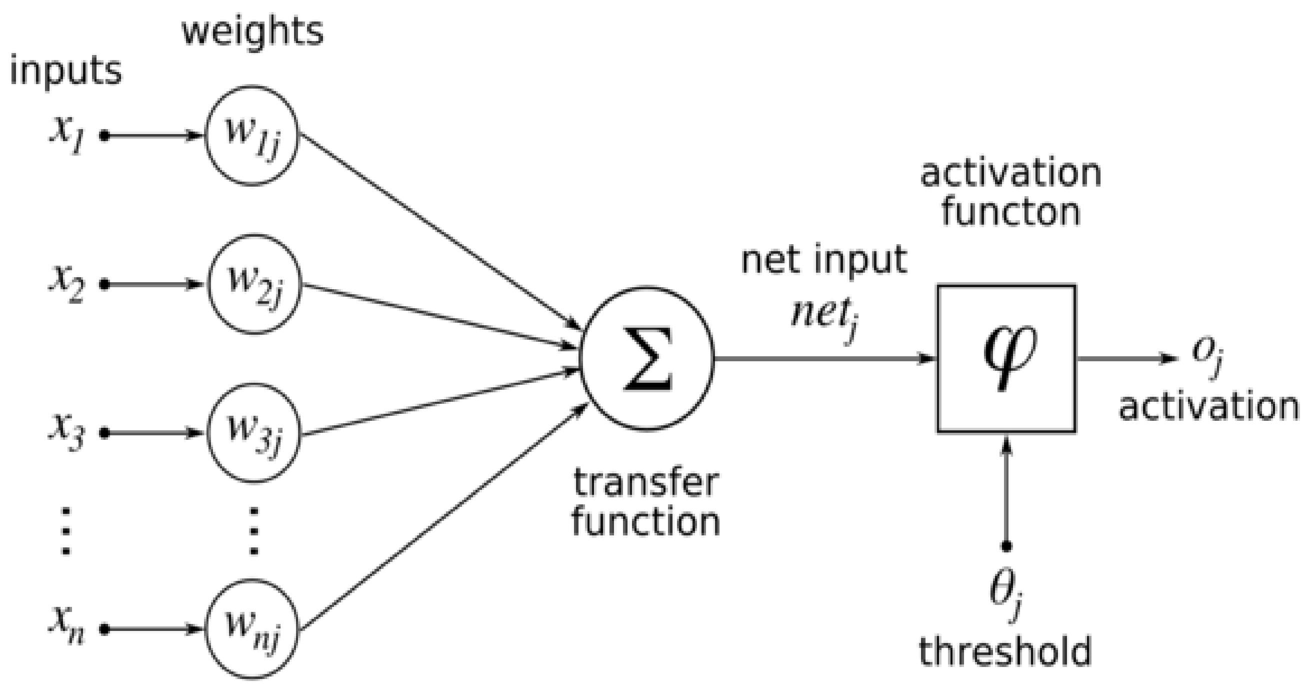


Inspiration for Neural Learning

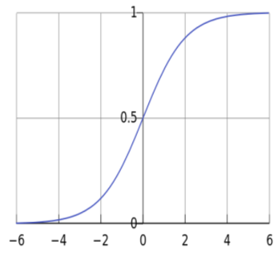
Neural Model



Neuron a.k.a. Unit

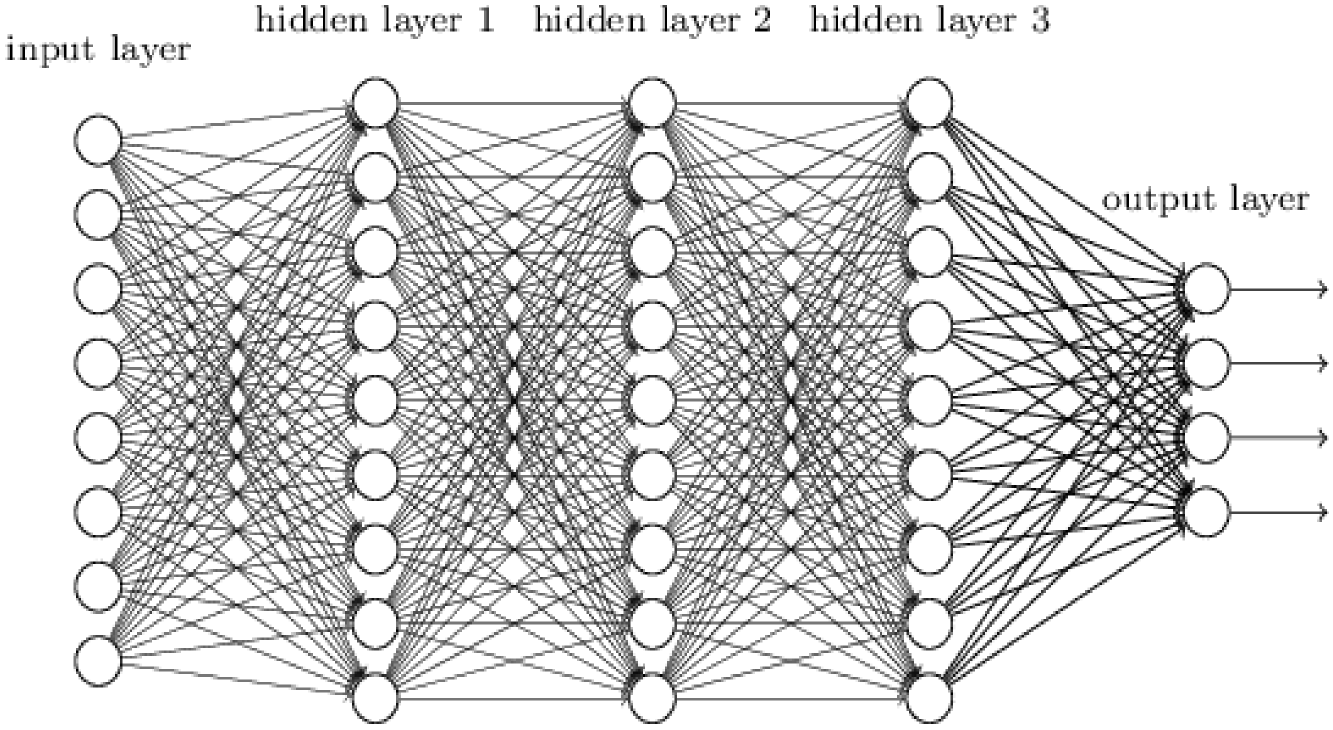


$$\phi = \frac{1}{1 + e^{-\Sigma}}$$

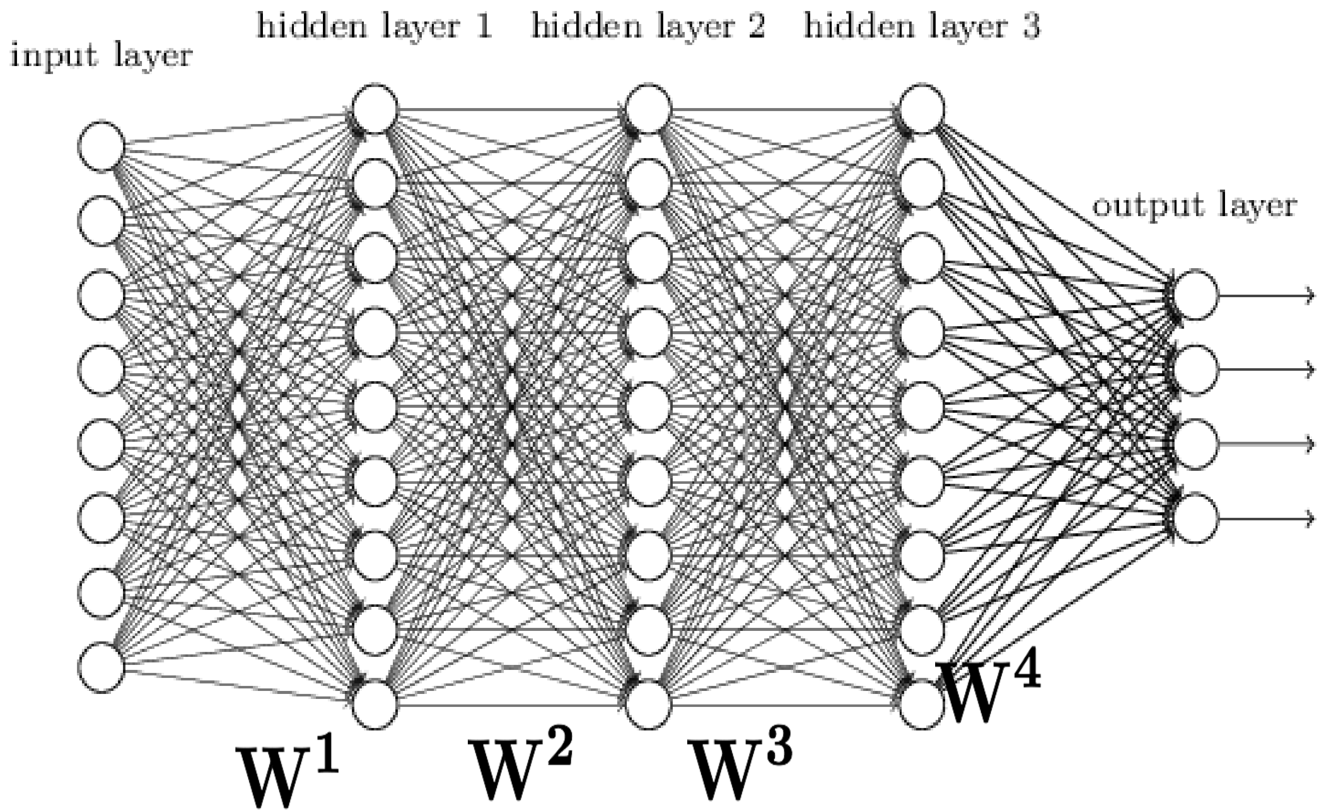


$$P(o_j = 1|x) = \phi\left(\sum_{i=1}^{n} w_{ij}x_i + \theta_j\right)$$

Feedforward Multilayered Network



Learning



input layer
hidden layer 1
hidden layer 2
hidden layer 3
output layer
$\mathbf{W}^1$
$\mathbf{W}^2$
$\mathbf{W}^3$
$\mathbf{W}^4$

Stochastic Gradient Descent

- Generalization of (Stochastic) Gradient Descent

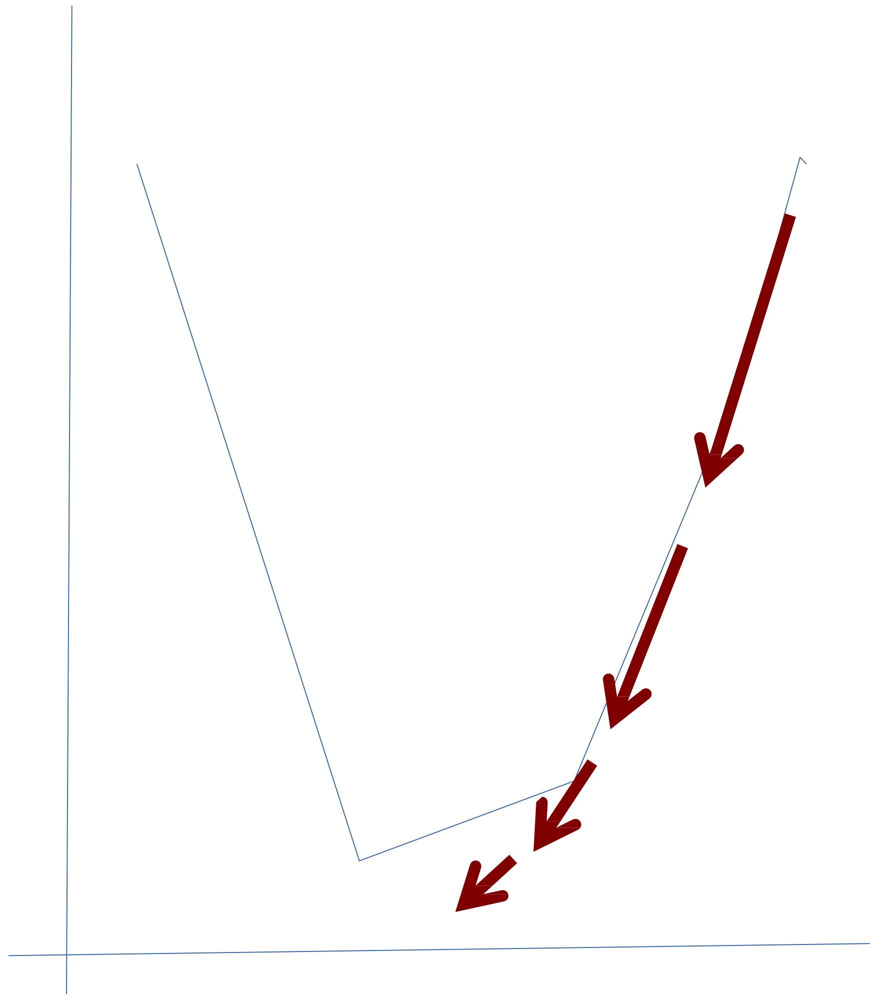
$$E = \frac{1}{2}(f - y)^2$$

$$f = \mathbf{w}^T \mathbf{x}$$

$$for \quad i = 1, 2, \ldots, n$$

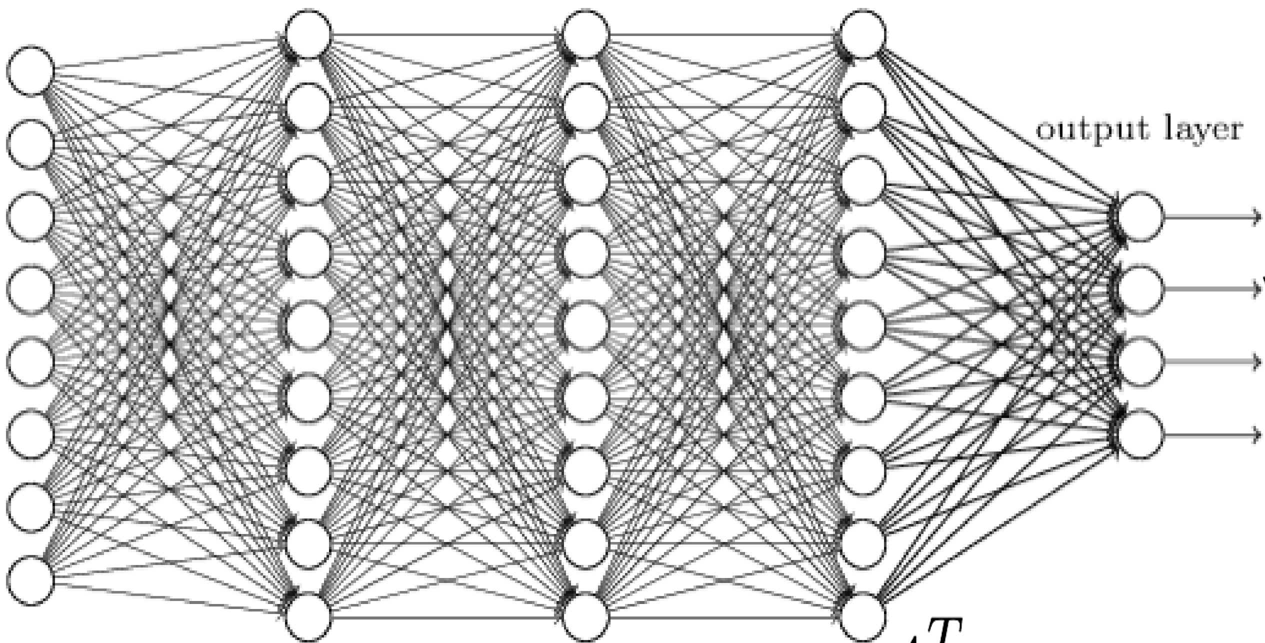
$$\mathbf{w} := \mathbf{w} - \eta \nabla_f E \mathbf{x}_i$$

Stochastic Gradient Descent



Backpropagation

$$F(x) := \sigma(\sigma(\mathbf{W^1}x)\mathbf{W^2}\dots)$$



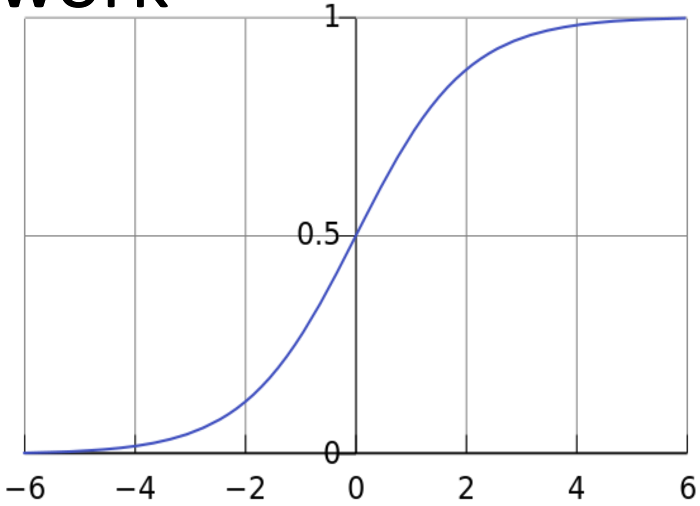
$$\nabla_f E$$

$${W^4}^T \nabla_f E$$

$${W^3}^T({W^4}^T \nabla_f E)$$

Backpropagation

- Does not work well in plain a normal” multilayer deep network
- Vanishing Gradients
- Slow Learning



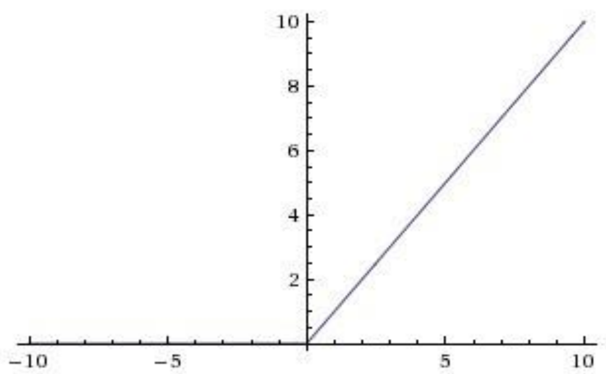
- SVM’s easier to train
- 2nd Neural Winter

Modern Deep Networks

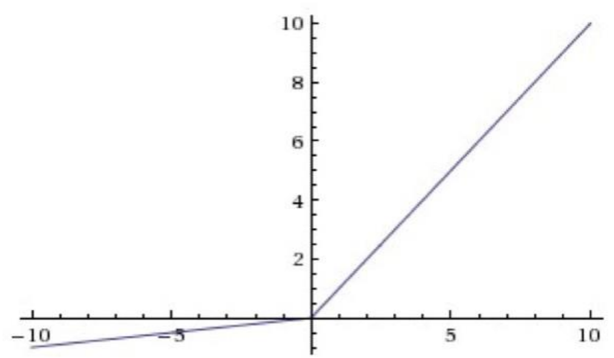
- Ingredients:

- Rectified Linear Activation function a.k.a. ReLu

$$\sigma(x) = max(0, x)$$



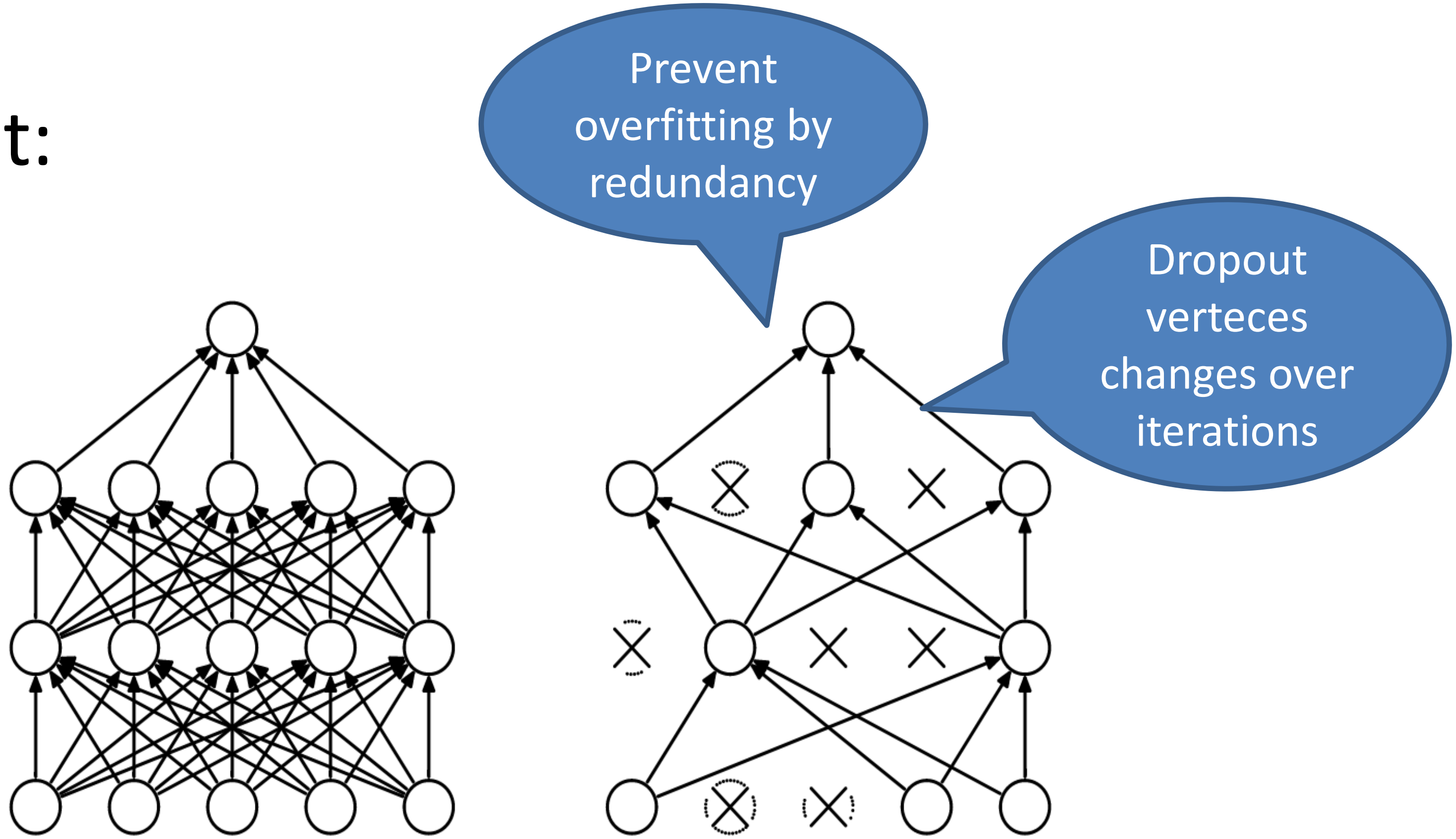
$$\sigma(x) = max(\alpha x, x) \quad \alpha < 1$$



Modern Deep Networks

- Ingredients:

- Dropout:



(a) Standard Neural Net (b) After applying dropout.

Modern Deep Networks

- Ingredients:

- Mini-batches:
  - Stochastic Gradient Descent
  - Compute gradient over many (50 -100) data points (minibatch) and update.

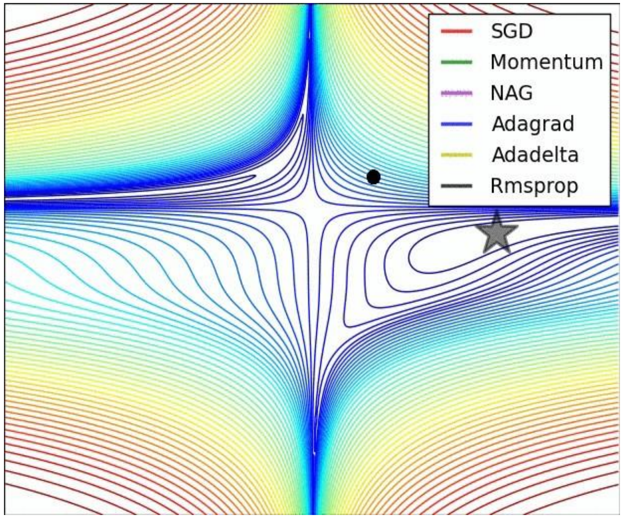
Modern Feedforward Networks

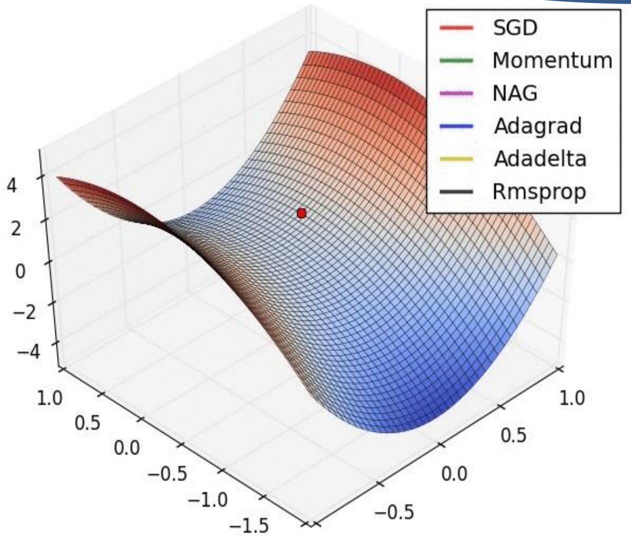
- Ingredients:
- Adagrad a.k.a. adaptive learning rates

$$\mathbf{w} := \mathbf{w} - \frac{\eta}{\sqrt{\sum_{j=0}^{j=i} \nabla_w E_j^2}} \nabla_f E \mathbf{x}_i$$

Decrease step size over time

In a factor-wise fashion





Deep Learning for RecSys

- Feature extraction directly from the content
  - Image, text, audio, etc.
  - Instead of metadata
  - For hybrid algorithms
- Heterogenous data handled easily
- Dynamic/Sequential behaviour modeling with RNNs
- More accurate representation learning of users and items
  - Natural extension of CF & more
- RecSys is a complex domain
  - Deep learning worked well in other complex domains
  - Worth a try

Research directions in DL-RecSys

- As of 2017 summer, main topics:
  - Learning item embeddings
  - Deep collaborative filtering
  - Feature extraction directly from content
  - Session-based recommendations with RNN
- And their combinations

Best practices

- Start simple
  - Add improvements later
- Optimize code
  - GPU/CPU optimizations may differ
- Scalability is key
- Opensource code
- Experiment (also) on public datasets
- Don’t use very small datasets
- Don’t work on irrelevant tasks, e.g. rating prediction

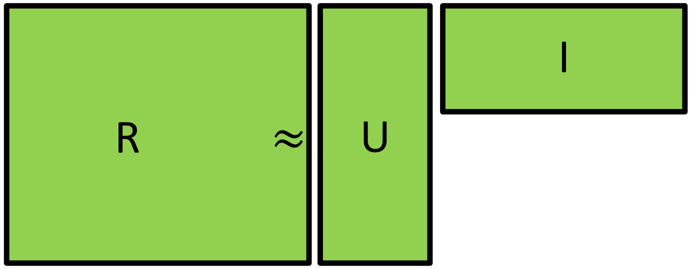
Item embeddings & 2vec models

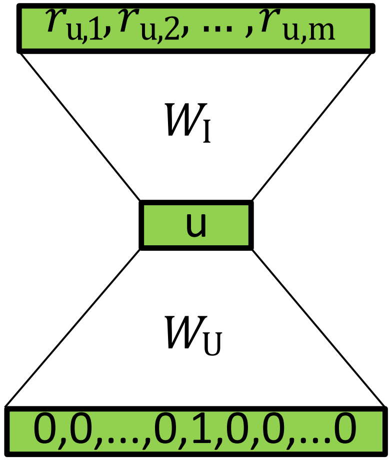
Embeddings

- Embedding: a (learned) real value vector representing an entity
  - Also known as:
    - Latent feature vector
    - (Latent) representation
  - **Similar entities’ embeddings are similar**
- Use in recommenders:
  - Initialization of item representation in more advanced algorithms
  - **Item-to-item recommendations**

Matrix factorization as learning embeddings

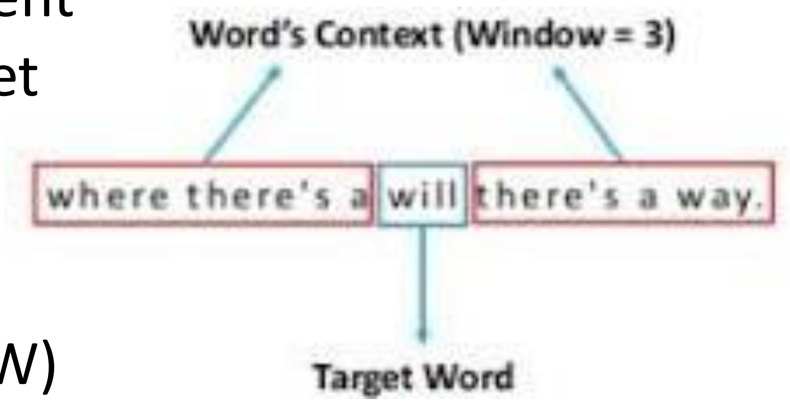
- MF: user & item embedding learning
  - Similar feature vectors
    - Two items are similar
    - Two users are similar
    - User prefers item
  - MF representation as a simplictic neural network
    - Input: one-hot encoded user ID
    - Input to hidden weights: user feature matrix
    - Hidden layer: user feature vector
    - Hidden to output weights: item feature matrix
    - Output: preference (of the user) over the items





Word2Vec

- [Mikolov et. al, 2013a]
- Representation learning of words
- Shallow model
- Data: (target) word + context pairs
  - Sliding window on the document
  - Context = words near the target
    - In sliding window
    - 1-5 words in both directions
- Two models
  - Continous Bag of Words (CBOW)
  - Skip-gram

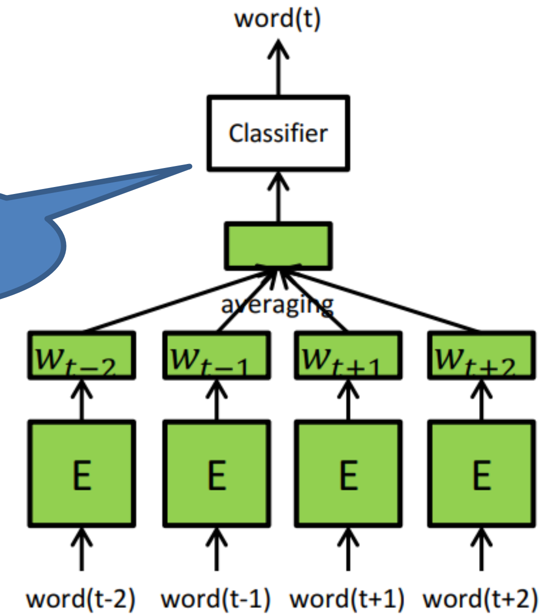


Word2Vec - CBOW

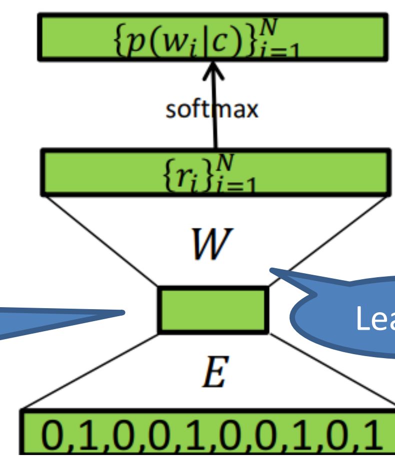
- Continuous Bag of Words
- Maximalizes the probability of the target word given the context
- Model
  - Input: one-hot encoded words
  - Input to hidden weights
    - Embedding matrix of words
  - Hidden layer
    - Sum of the embeddings of the words in the context
  - Hidden to output weights
  - Softmax transformation
    - Smooth approximation of the max operator
    - Highlights the highest value

$$s_i = \frac{e^{r_i}}{\sum_{j=1}^{N} e^{r_j}}, \ (r_j\text{: scores})$$

  - Output: likelihood of words of the corpus given the context
- Embeddings are taken from the input to hidden matrix
  - Hidden to output matrix also has item representations (but not used)



„Fake“ Prediction



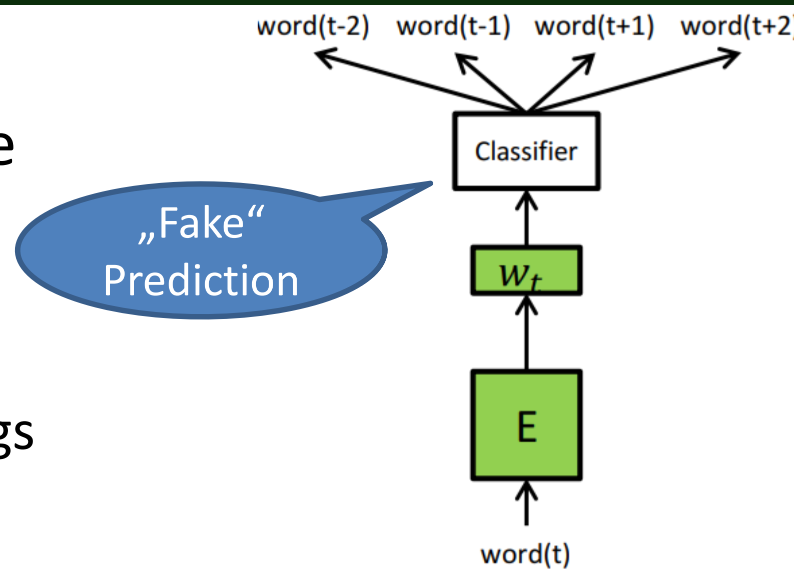
Learning

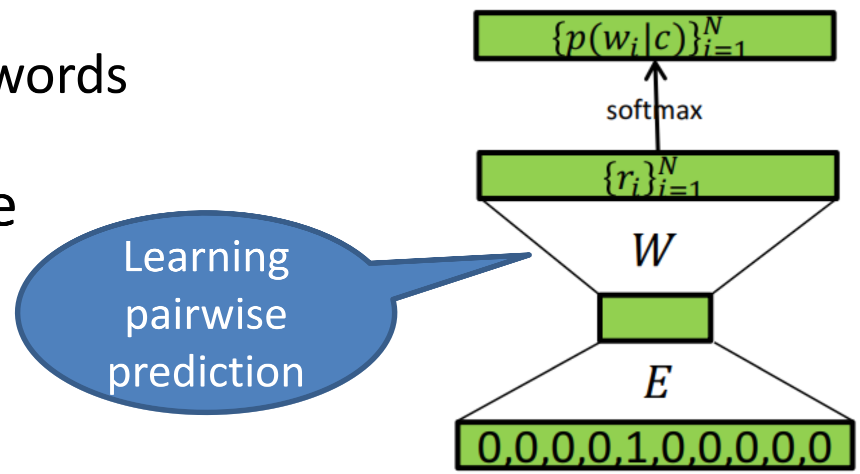
Our TRUE target - embedings

Shared input/output weights for thesame words

Word2Vec – Skip-gram

- Maximalizes the probability of the context, given the target word
- Model
  - Input: one-hot encoded word
  - Input to hidden matrix: embeddings
  - Hidden state
    - Item embedding of target
  - Softmax transformation
  - Output: likelihood of context words (given the input word)
- Reported to be more accurate



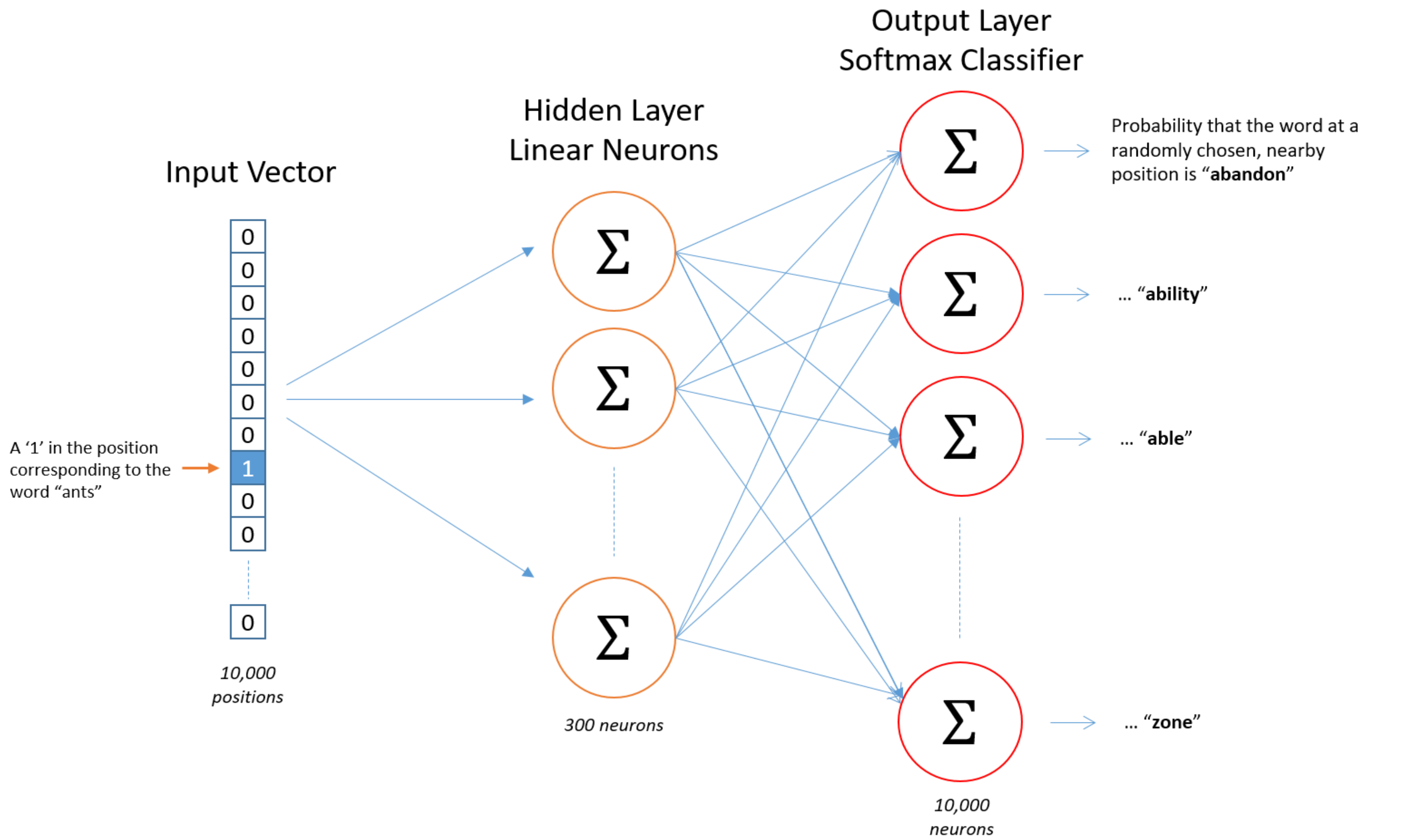


Word2Vec – Skip-gram

Source Text	Training Samples
The quick brown fox jumps over the lazy dog.	(the, quick) (the, brown)
The quick brown fox jumps over the lazy dog.	(quick, the) (quick, brown) (quick, fox)
The quick brown fox jumps over the lazy dog.	(brown, the) (brown, quick) (brown, fox) (brown, jumps)
The quick brown fox jumps over the lazy dog.	(fox, quick) (fox, brown) (fox, jumps) (fox, over)

http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

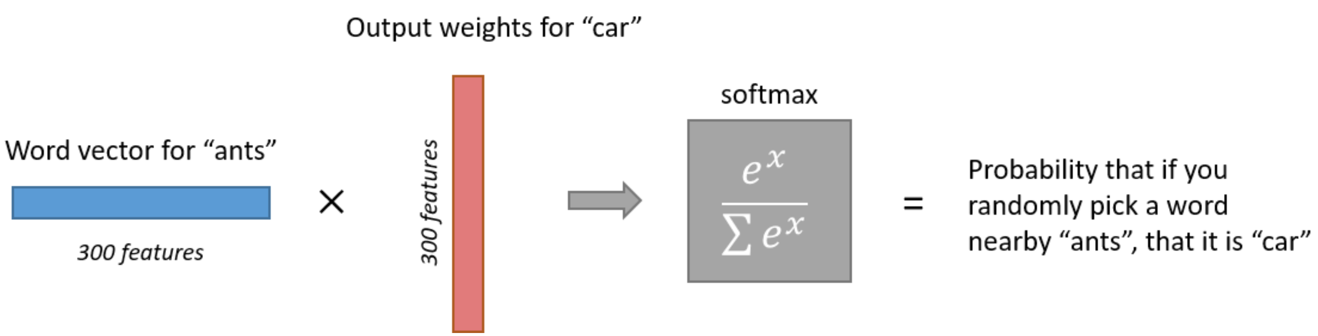
Word2Vec – Skip-gram



Word2Vec – Skip-gram

- Shared weights for the same input and output word
- Afterwards, apply softmax to get a probability distribution

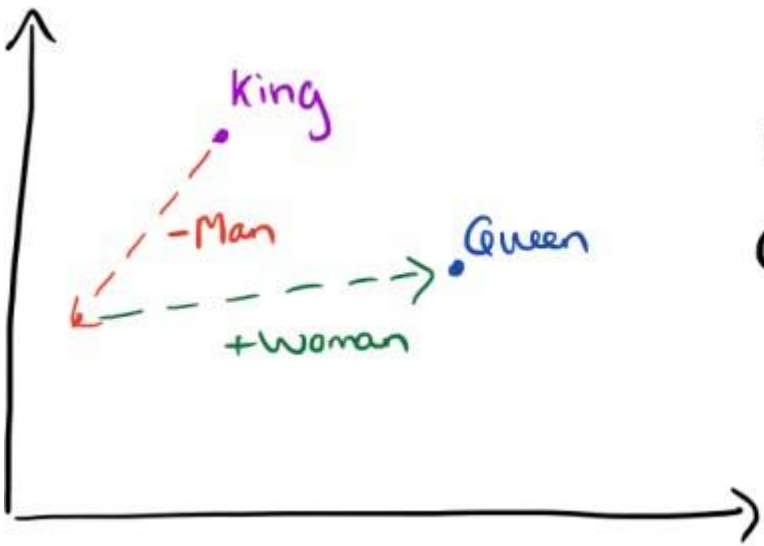
- Still, too many weights -> sample negative elements to be updated
    - Negative sampling, http://mccormickml.com/2017/01/11/word2vec-tutorial-part-2-negative-sampling/



http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

Geometry of the Embedding Space

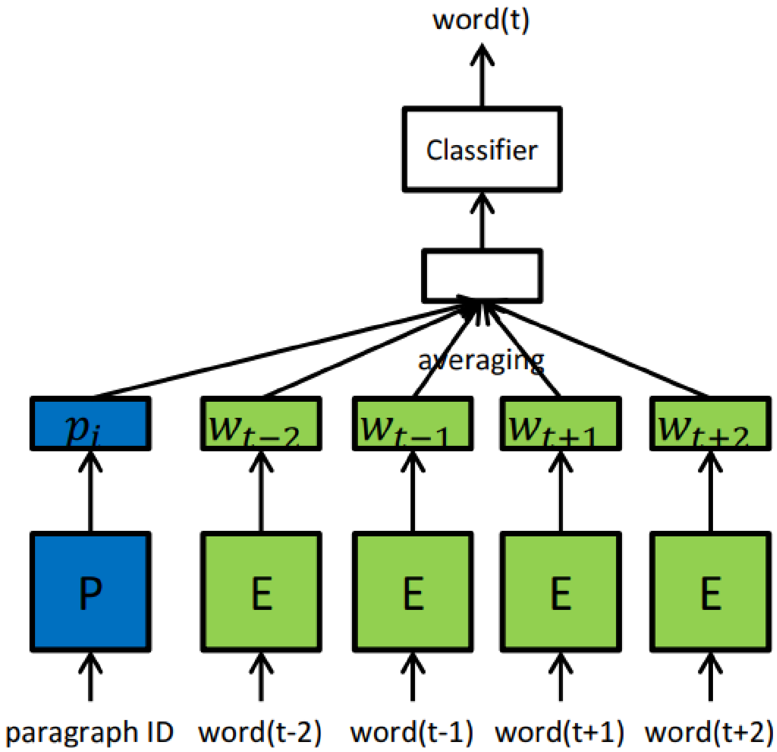
King - Man + Woman = Queen



Vector
Composition

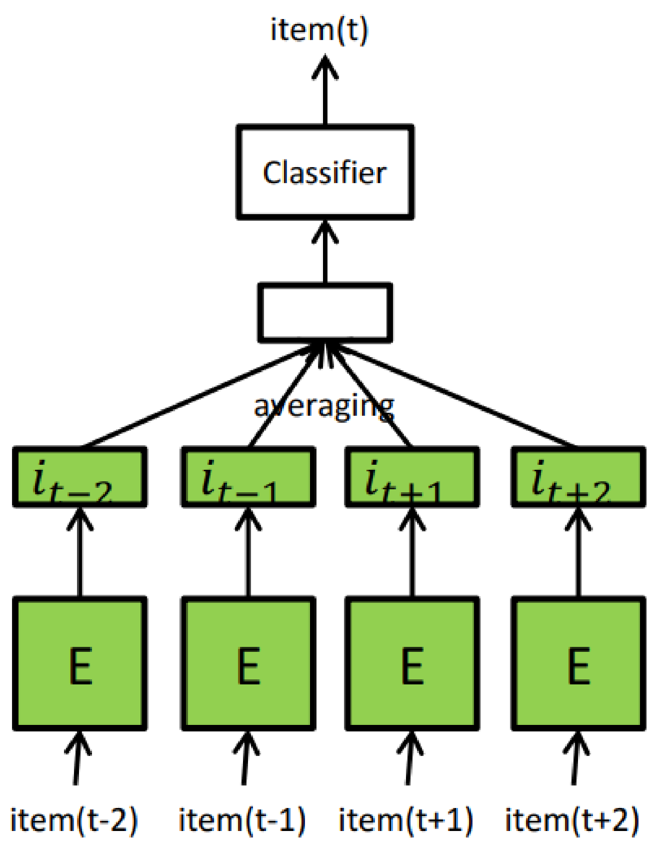
Paragraph2vec, doc2vec

- [Le & Mikolov, 2014]
- Learns representation of paragraph/document
- Based on CBOW model
- Paragraph/document embedding added to the model as global context



...2vec for Recommendations

Replace words with items in a session/user profile

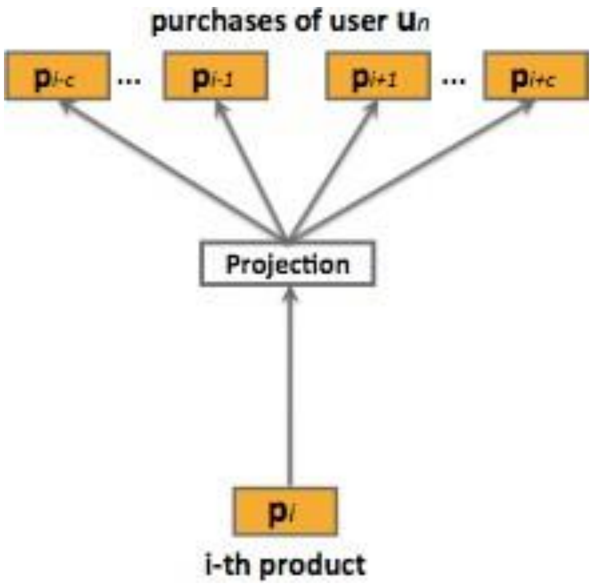


Prod2Vec

- [Grbovic et. al, 2015]
- Skip-gram model on products
  - Input: i-th product purchased by the user
  - Context: the other purchases of the user
- Bagged prod2vec model
  - Input: products purchased in one basket by the user
    - Basket: sum of product embeddings
  - Context: other baskets of the user
- Learning user representation
  - Follows paragraph2vec
  - User embedding added as global context
  - Input: user + products purchased except for the i-th
  - Target: i-th product purchased by the user
- [Barkan & Koenigstein, 2016] proposed the same model later as item2vec
  - Skip-gram with Negative Sampling (SGNS) is applied to event data

Prod2Vec

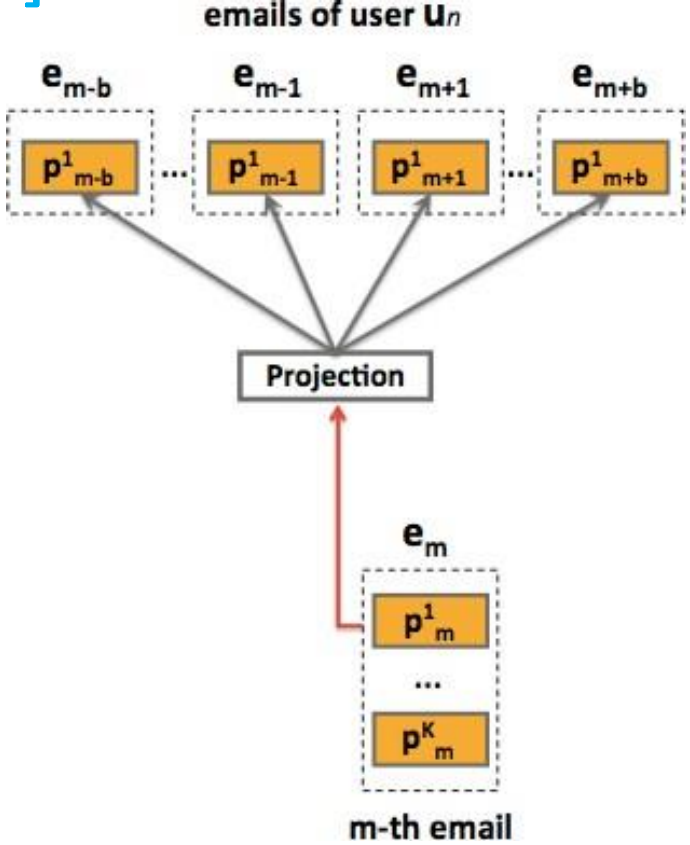
[Grbovic et. al, 2015]



pro2vec skip-gram model on products

Bagged Prod2Vec

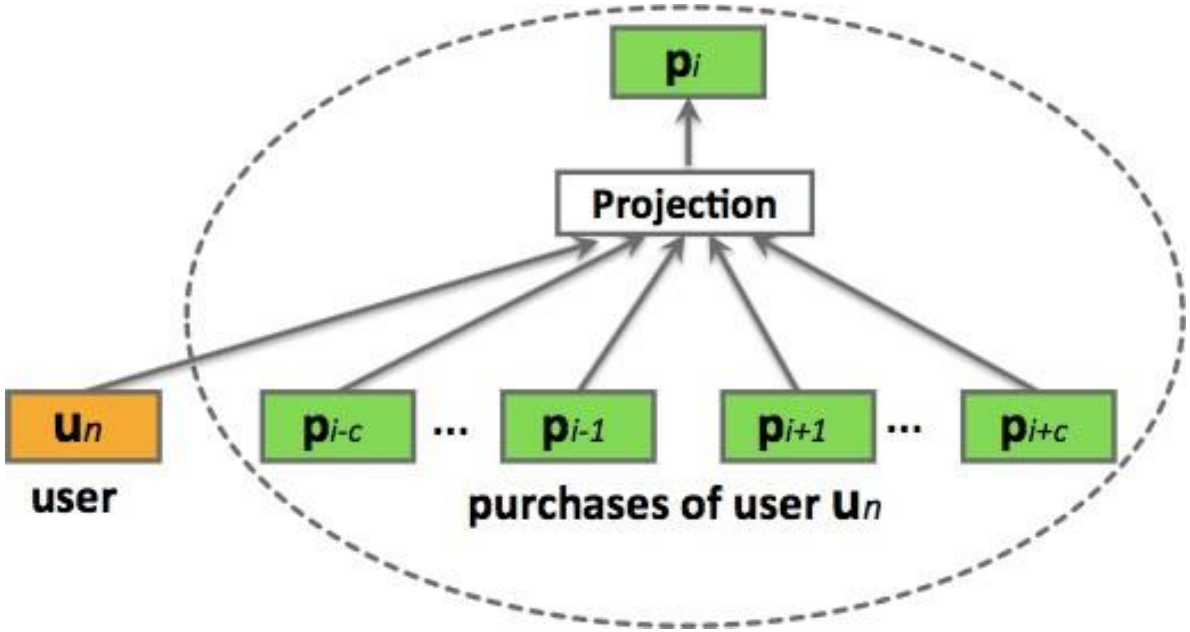
[Grbovic et. al, 2015]



bagged-prod2vec model updates

User-Prod2Vec

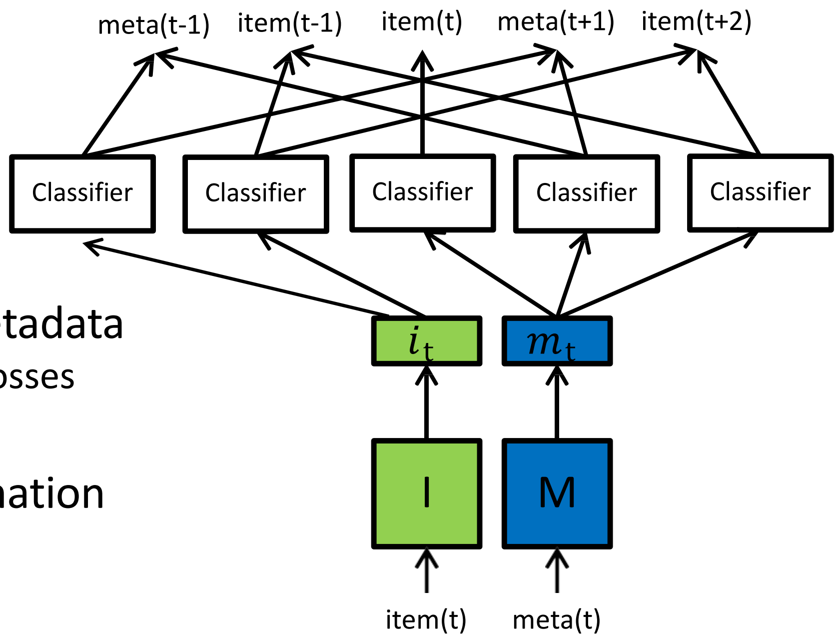
[Grbovic et. al, 2015]



User embeddings for user to product predictions

Utilizing more information

- Meta-Prod2vec [Vasile et. al, 2016]
  - Based on the prod2vec model
  - Uses item metadata
    - Embedded metadata
    - Added to both the input and the context
  - Losses between: target/context item/metadata
    - Final loss is the combination of 5 of these losses
- Content2vec [Nedelec et. al, 2017]
  - Separate modules for multimodel information
    - CF: Prod2vec
    - Image: AlexNet (a type of CNN)
    - Text: Word2Vec and TextCNN
  - Learns pairwise similarities
    - Likelihood of two items being bought together



References

- [Barkan & Koenigstein, 2016] O. Barkan, N. Koenigstein: ITEM2VEC: Neural item embedding for collaborative filtering. IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP 2016).
- [Grbovic et. al, 2015] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, D. Sharp: E-commerce in Your Inbox: Product Recommendations at Scale. 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'15).
- [Le & Mikolov, 2014] Q. Le, T. Mikolov: Distributed Representations of Sentences and Documents. 31st International Conference on Machine Learning (ICML 2014).
- [Mikolov et. al, 2013a] T. Mikolov, K. Chen, G. Corrado, J. Dean: Efficient Estimation of Word Representations in Vector Space. ICLR 2013 Workshop.
- [Mikolov et. al, 2013b] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean: Distributed Representations of Words and Phrases and Their Compositionality. 26th Advances in Neural Information Processing Systems (NIPS 2013).
- [Morin & Bengio, 2005] F. Morin, Y. Bengio: Hierarchical probabilistic neural network language model. International workshop on artificial intelligence and statistics, 2005.
- [Nedelec et. al, 2017] T. Nedelec, E. Smirnova, F. Vasile: Specializing Joint Representations for the task of Product Recommendation. 2nd Workshop on Deep Learning for Recommendations (DLRS 2017).
- [Vasile et. al, 2016] F. Vasile, E. Smirnova, A. Conneau: Meta-Prod2Vec – Product Embeddings Using Side-Information for Recommendations. 10th ACM Conference on Recommender Systems (RecSys'16).