Multi-Rate Deep Learning for Temporal Recommendation

Yang Song
Microsoft Research Redmond
yangsong@microsoft.com

Ali Mamdouh Elkahky
Columbia University
ame2154@columbia.edu

Xiaodong He
Microsoft Research Redmond
xiaohe@microsoft.com

SIGIR 2016

TDSSM User embedding

$$E(U, t_i) = f_{U,t_i}(E_{base}(U), E_{t_i}(U))$$

$$f(W_1, W_2) = \begin{cases} W_1 \odot W_2 & \text{elem-wise multiplication} \\ W_1 \odot A \odot W_2 & \text{weighted elem-wise multiplication} \\ [W_1; W_2] & \text{concatenation} \end{cases}$$

Objective function

$$\min_{W_{user} W_{item}} -\log \prod_{user, item^+, t_i} p(item^+ | user, t_i)$$

where $p(item^+|user, t_i)$ is defined as:

$$p(item^+|user, t_i) = \frac{e^{cos\left(E(user,t_i), E_{item^+}\right)}}{\sum_{\forall item} e^{cos(E(user,t_i), E_{item})}}$$

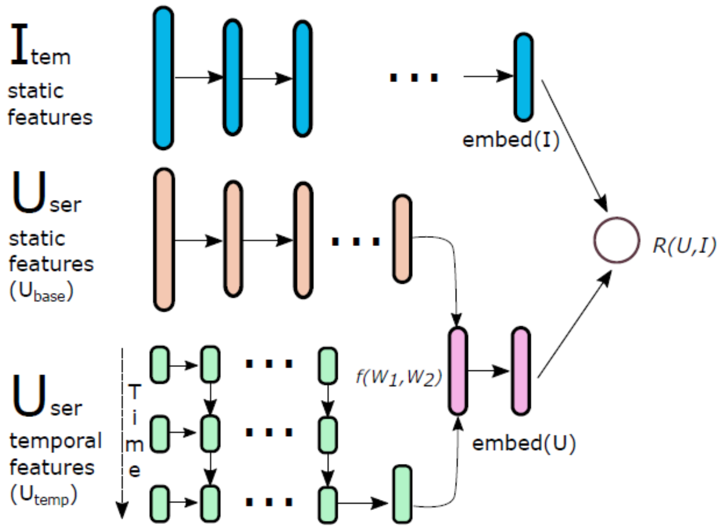


Figure 1: The temporal DSSM (TDSSM) model for temporal user modeling.

Mutli-Rate TDSSM

Fast-rate RNNs: very recent user interests
Slow-rate RNNs: model seasonal user interest shift

fully-connected feedfoward network

pre-train:

reduces the size of input feature space by a factor proportional to the ratio between the size of the original feature space (i.e. $|X_{user}| = D_{user}$ and $|X_{item}| = D_{item}$) to the size of the embedding space(i.e. $|E_{user}| = |E_{item}| = d$).

Modeling Interestingness with Deep Neural Networks

Jianfeng Gao, Patrick Pantel, Michael Gamon, Xiaodong He, Li Deng
Microsoft Research
One Microsoft Way
Redmond, WA 98052, USA
{jfgao,ppantel,mgamon,xiaohe,deng}@microsoft.com

EMNLP 2014

Input Layer

represent w by a one-hot vector (one-hot encoder)
map w to a separate tri-letter vector
("#dog#" -> "#do", "dog", and "og#")

Convolutional Layer

$$\mathbf{u}_i = \tanh\left(\mathbf{W}_c^{\mathrm{T}} \mathbf{c}_i\right), \text{where } i = 1 \ldots I$$

Max-pooling Layer

$$v(j) = \max_{i=1,\ldots,I} \{\mathrm{u}_i(j)\} \qquad (2)$$

where the max operation is performed for each dimension of $\mathbf{u}$ across $i = 1, \ldots, I$ respectively.

Fully-Connected Layers

$$\mathbf{h} = \tanh(\mathbf{W}_1^{\mathrm{T}} \mathbf{v})$$

$$\mathbf{y} = \tanh(\mathbf{W}_2^{\mathrm{T}} \mathbf{h})$$

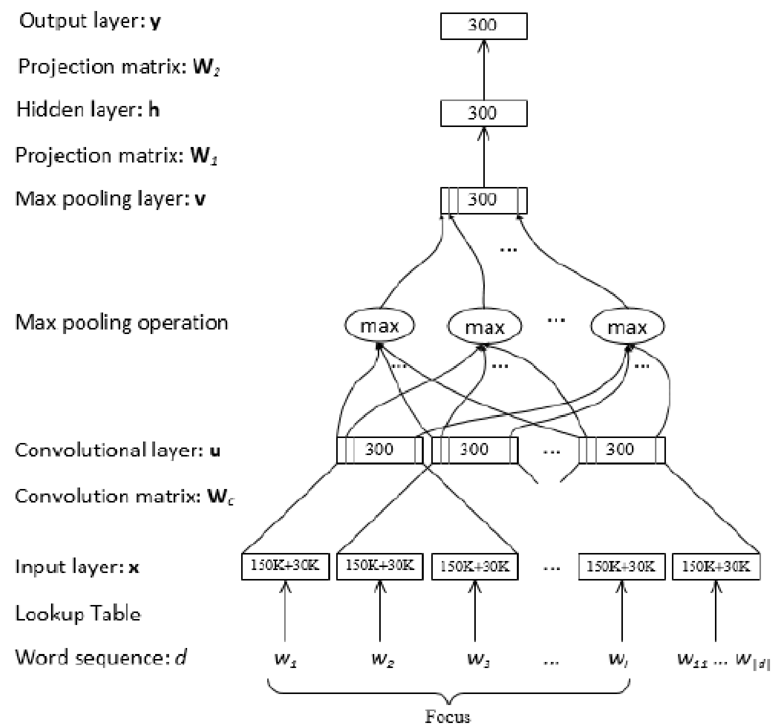


Figure 1: Illustration of the network architecture and information flow of the DSSM

Training the DSSM

$$\sigma(s,t) \equiv \mathrm{sim}_{\boldsymbol{\theta}}(s,t) = \frac{\mathbf{y}_s^{\mathrm{T}}\mathbf{y}_t}{\|\mathbf{y}_s\|\|\mathbf{y}_t\|}$$

a pair-wise rank loss

construct two pairs of documents (s, t_1) and (s, t_2)

Δ be the difference of their interestingness scores: $\Delta = \sigma(s,t_1) - \sigma(s,t_2)$

logistic loss:

$$\mathcal{L}(\Delta; \boldsymbol{\theta}) = \log(1 + \exp(-\gamma\Delta))$$

ConTagNet: Exploiting User Context for Image Tag Recommendation

Yogesh Singh Rawat
School of Computing
National University of Singapore
yogesh@comp.nus.edu.sg

Mohan S Kankanhalli
School of Computing
National University of Singapore
mohan@comp.nus.edu.sg

MM 2016

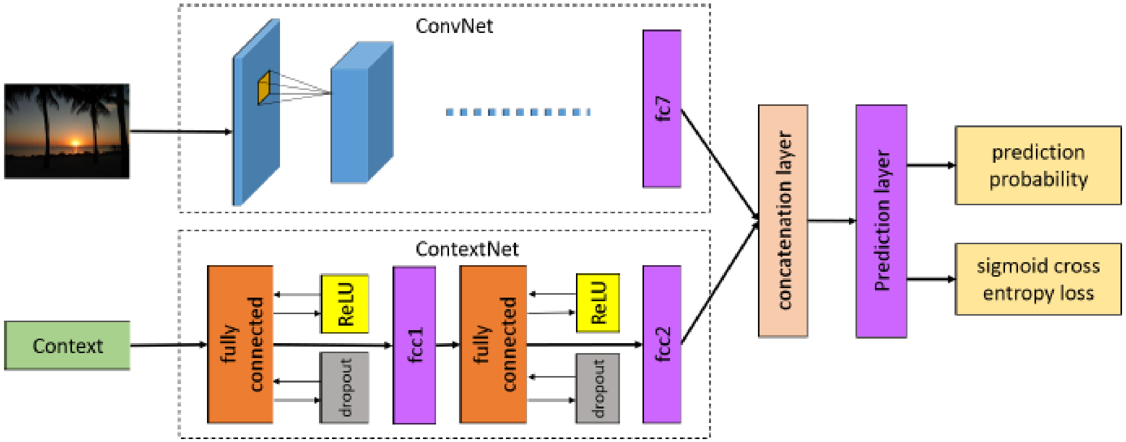


Figure 2: Overview of the proposed model. An input image is passed through a CNN (ConvNet) component of the network which has been adopted from AlexNet [15]. The associated context of the image is processed through a parallel NN (ContextNet). The output of ConvNet (fc7 layer) and ContextNet (fcc2 layer) are merged together and passed to the prediction layer for multi-label prediction. The proposed model can be trained end-to-end using gradient descent optimization.

图片的特征提取选择AlexNet网络

meta-data (时间、位置) 2-layered neural network with fully connected layers

两个网络结果串联，a fully connected layer

网络训练（sigmoid cross-entropy）

$$L_e = -\frac{1}{N}\sum_{i=1}^{N}[p_i log(\hat{p}_i) + (1-p_i)log(1-\hat{p}_i)] \qquad (2)$$

Fidelity-Weighted Learning

Mostafa Dehghani
University of Amsterdam
dehghani@uva.nl

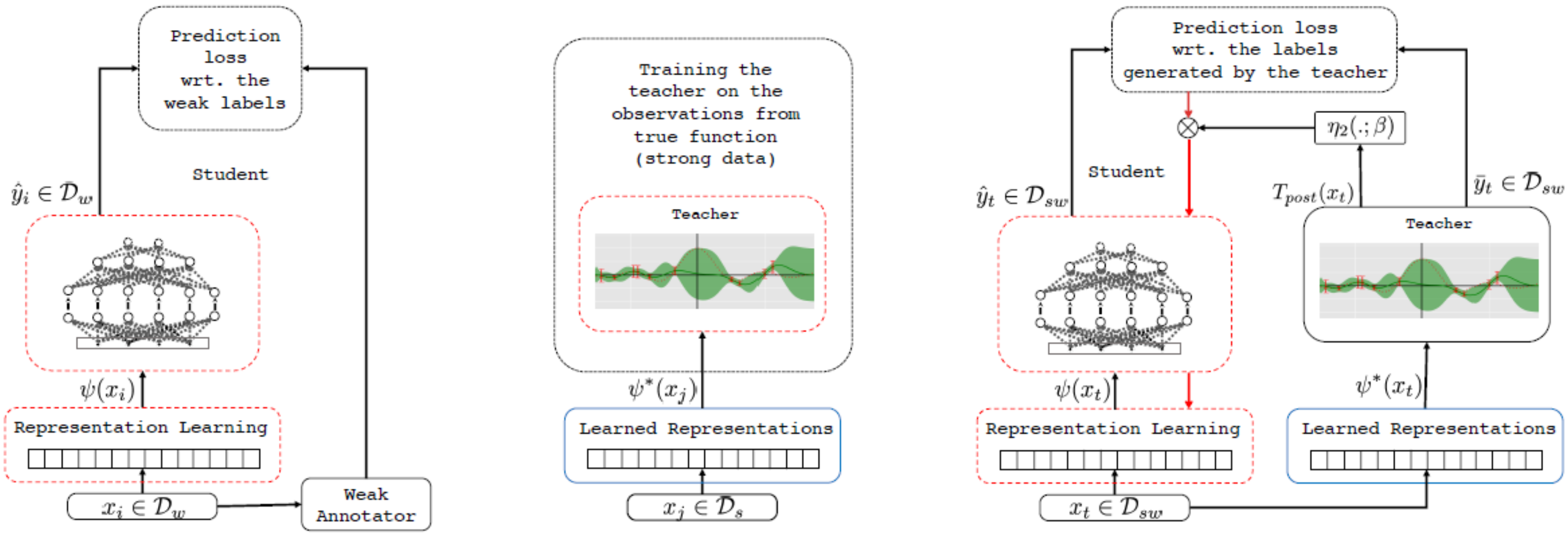
Arash Mehrjou
MPI for Intelligent Systems
amehrjou@tuebingen.mpg.de

Stephan Gouws
Google Brain
sgouws@google.com

Jaap Kamps
University of Amsterdam
kamps@uva.nl

Bernhard Schölkopf
MPI for Intelligent Systems
bs@tuebingen.mpg.de

ICLR 2018



(a) Step 1 (b) Step 2 (c) Step 3

Figure 1: Illustration of Fidelity-Weighted Learning: Step 1: Pre-train student on weak data, Step 2: Fit teacher to observations from the true function, and Step 3: Fine-tune student on labels generated by teacher, taking the confidence into account. Red dotted borders and blue solid borders depict components with trainable and non-trainable parameters, respectively.