Cold-Start Aware User and Product Attention for Sentiment Classification

Reinald Kim Amplayo, Jihyeok Kim, Sua Sung, Seung-won Hwang
Yonsei University

The Problem

Sentiment Classification

- Movie reviews (IMDB)

User: mat***dock

Not much to say it is just boring. And I have watched Incredibles 1 so many times since when I was 10 years old, never got bored. This movie was boring and had too many weak characters. The villain twist was very obvious and the cringey sjw moments ruined the feel of the movie.

→ 3/10

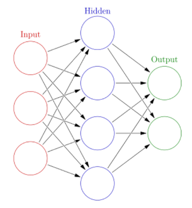
- Restaurant reviews (Yelp)

User: Ran*** H.

Definitely eat here. Delicious food. I can't say much that hasn't already been said in other reviews that lead us to come here. The staff were spectacularly attentive and kind. The duck larb is just next level. We also came back here despite only having 48 hours in Melbourne.

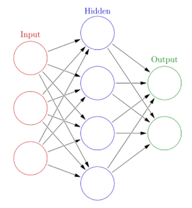
→ 5/5

Traditional Methods: Use **only text** as features

Not much to say it is just boring. And I have watched Incredibles 1 so many times since ... →  → 3/10

New Methods: **Add User/Product Information!**

User: mat***dock

\+ Not much to say it is just boring. And I have watched Incredibles 1 so many times since ... →  → 3/10

- Some expressions are user- or product-specific

The food is ***very salty****!* – may have different sentiments for different users

Existing Solutions

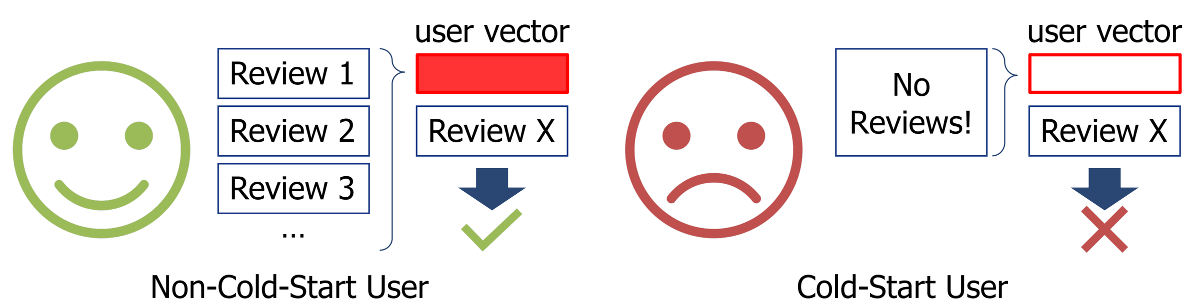
Focused on "**where** do we add these information?"

- **UPNN:** Preference matrix to modify word meaning!
- **UPDMN:** Memory networks and modify document meaning!
- **NSC**: Attention mechanism to modify either/both sentence and/or document meaning!

Result: **Attention mechanism** is the best location!

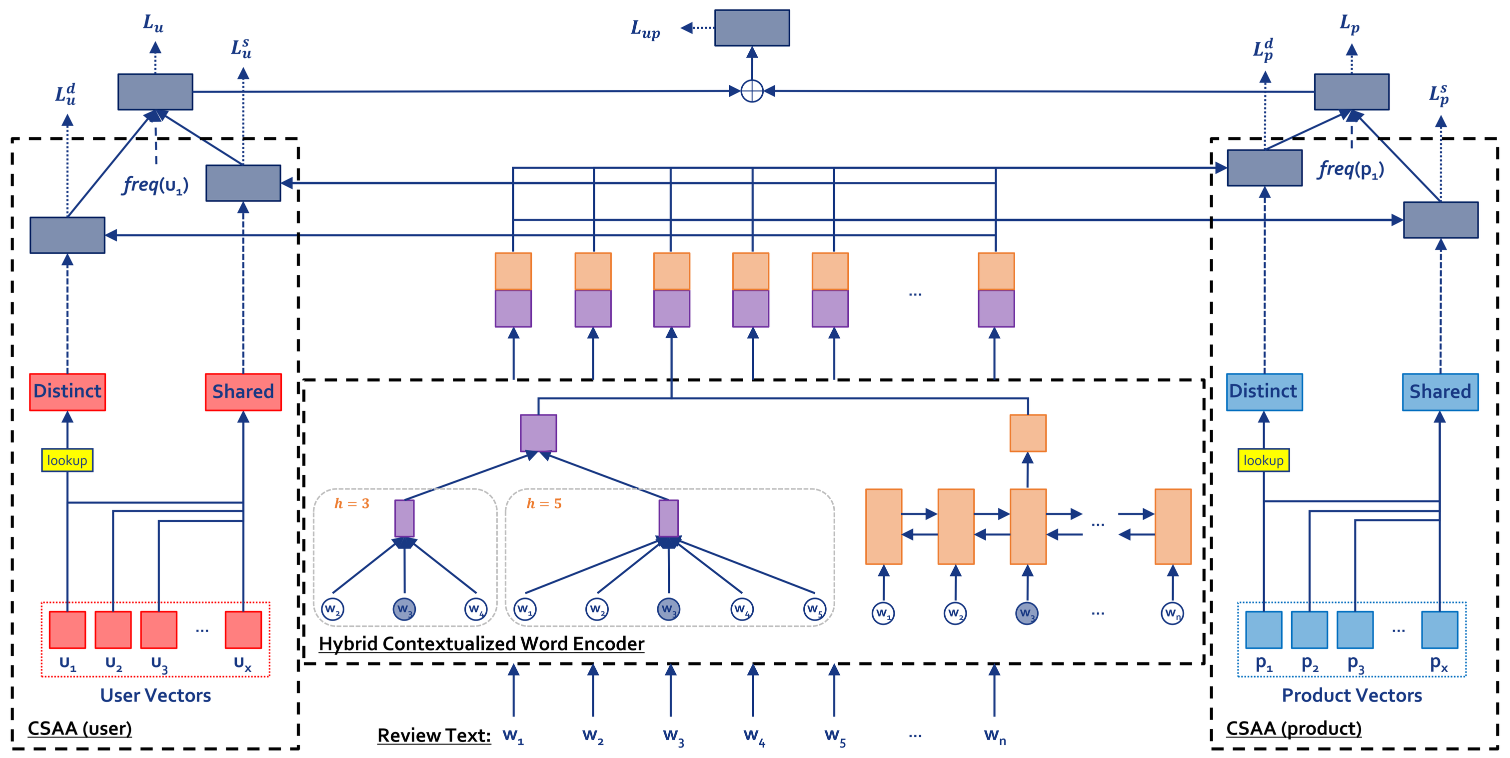
Bigger Problem:

How about **cold-start** users/products?



- Naively using user/product information leads to **incorrectly trained vectors**

Hybrid Contextualized Sentiment Classifier (HCSC)



Hybrid Contextualized Word Encoder

- Previous models used complicated models (e.g. hierarchical models) to improve performance with a price of **slower training**

Idea: simple model + cold-start aware = **fast training** + **improve performance**

Intuition: use both **local (CNN)** and **global (BiLSTM)** to contextualized words and simply concatenate the results

$$w_{i,cnn} = \mathrm{CNN}(w_i)$$
$$w_{i,rnn} = \mathrm{BiLSTM}(w_i)$$
$$\widehat{w}_i = [w_{i,cnn}; w_{i,rnn}]$$

Cold-Start Aware Attention (CSAA)

Distinct pooled vectors

$$e_u^d(w_i, u) = v^{d\top} \tanh\left(W_w^d w_i + W_u^d u + b^d\right)$$
$$a_{u,i}^d = \mathrm{softmax}(e_u^d(w_i, u))$$
$$\boldsymbol{v}_u^d = \sum a_{u,i}^d * w_i$$

If user is not cold-start, use the ***original attention mechanism****.*

Shared pooled vectors

$$e_u^s(\mu(\{w_i\}), u_k) = \mu(\{w_i\}) W_u^s u_k$$
$$a_{u,k}^s = \mathrm{softmax}\left(e_{u,k}^s(\mu(\{w_i\})), u_k\right)$$
$$u' = a_{u,k}^s * u_k$$
$$\boldsymbol{v}_u^s = \mathrm{attention}(\mu(\{w_i\}), u')$$

If user is cold-start, create a ***pseudo-user vector*** *through other users. Use that in the attention mechanism.*

Frequency-guided selective gate

$$g_u = 1 - \exp\left(-\left(f(u)/relu(\lambda_u)\right)^{relu(k_u)}\right)$$
$$\boldsymbol{v}_u = g_u * v_u^d + (1 - g_u) * v_u^s$$

Select between two pooled vectors using ***Weibull distribution*** *controlled by user review frequency*

Repeat for product information to get $\boldsymbol{v}_p^d$, $\boldsymbol{v}_p^s$, and $\boldsymbol{v}_p$!

Sentiment Classification

1. Combine $\boldsymbol{v}_u$ and $\boldsymbol{v}_p$ using a sigmoidal gate

$$g_{up} = \sigma\left(W_g[v_u; v_p] + b_g\right)$$
$$v_{up} = g_{up} * v_u + (1 - g_{up}) * v_p$$

2. Create a softmax classifier to be used by all the pooled vectors
3. Train the classifier using cross-entropy loss of all pooled vectors

$$\mathbb{V} = \{v_u^d, v_u^s, v_p^d, v_p^s, v_u, v_p, v_{up}\}$$
$$y'(v) = softmax(Wv + b)$$
$$L = -\sum_{d \in D} \sum_{c \in C} \sum_{v \in \mathbb{V}} y_c^{(d)} * \log\left(y_c^{\prime(d)}(v)\right)$$

4. At test time, use $y'(v_{up})$ to classify sentiments

Source code available here:

Q1: How do models perform on **sparse data**?

Models	IMDB		Yelp 2013	
	Acc.	RMSE	Acc.	RMSE
JMARS	-	1.773*	-	0.985*
UPNN	0.435*	1.602*	0.596*	0.784*
TLFM+PRC	-	1.352*	-	0.716*
UPDMN	0.465*	1.351*	0.639*	0.662
TUPCNN	0.488*	1.451*	0.639*	0.694*
NSC	0.533	1.281*	0.650	0.692*
CNN+CSAA	0.522*	1.256*	0.654	0.665
RNN+CSAA	0.527*	1.237*	0.654	0.667
HCSC	**0.542**	**1.213**	**0.657**	**0.660**

Models	Sparse20	Sparse50	Sparse80
NSC(LA)	0.469	0.428	0.309
NSC	0.497	0.408	0.292
CNN+CSAA	0.497	0.444	0.343
RNN+CSAA	**0.505**	0.455	0.364
HCSC	**0.505**	**0.456**	**0.368**

(a) IMDB Datasets

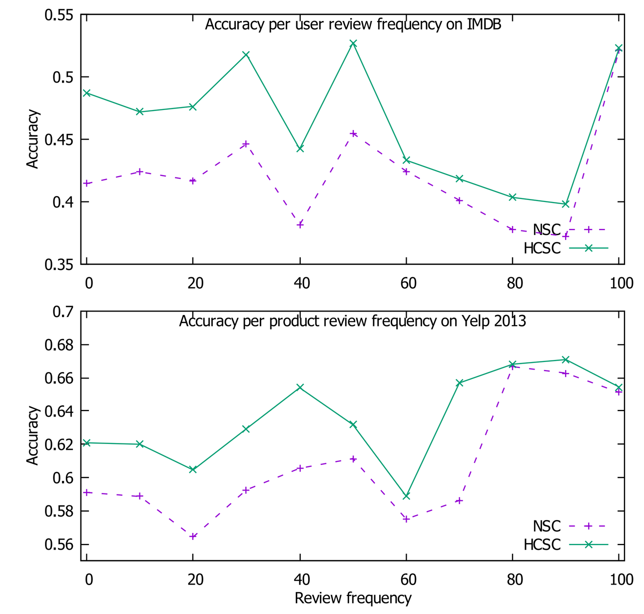
Models	Sparse20	Sparse50	Sparse80
NSC(LA)	0.624	0.590	0.523
NSC	0.626	0.592	0.511
CNN+CSAA	0.626	0.605	0.522
RNN+CSAA	0.633	0.603	0.527
HCSC	**0.636**	**0.608**	**0.538**

(b) Yelp 2013 Datasets

- By solving the cold-start problem, HCSC achieves **better results** without sacrificing training speed
- On sparse data with more cold-start users/products,

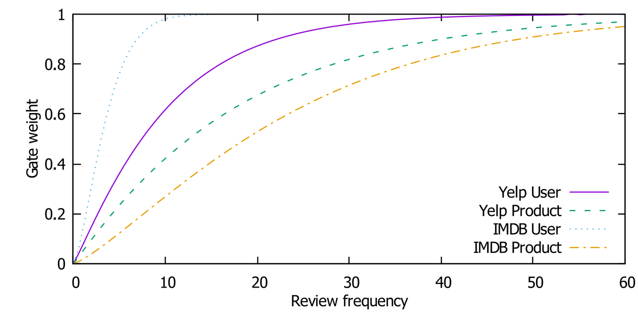
1. Previous SOTA (i.e. NSC) performs worse than the same model without using user/product information (i.e. NSC(LA))
2. HCSC **still performs well** even in cold-start conditions

Q2: When does HCSC get **accuracy gains**?



- When review frequency is **smaller**, increase in performance is seen from NSC to HCSC

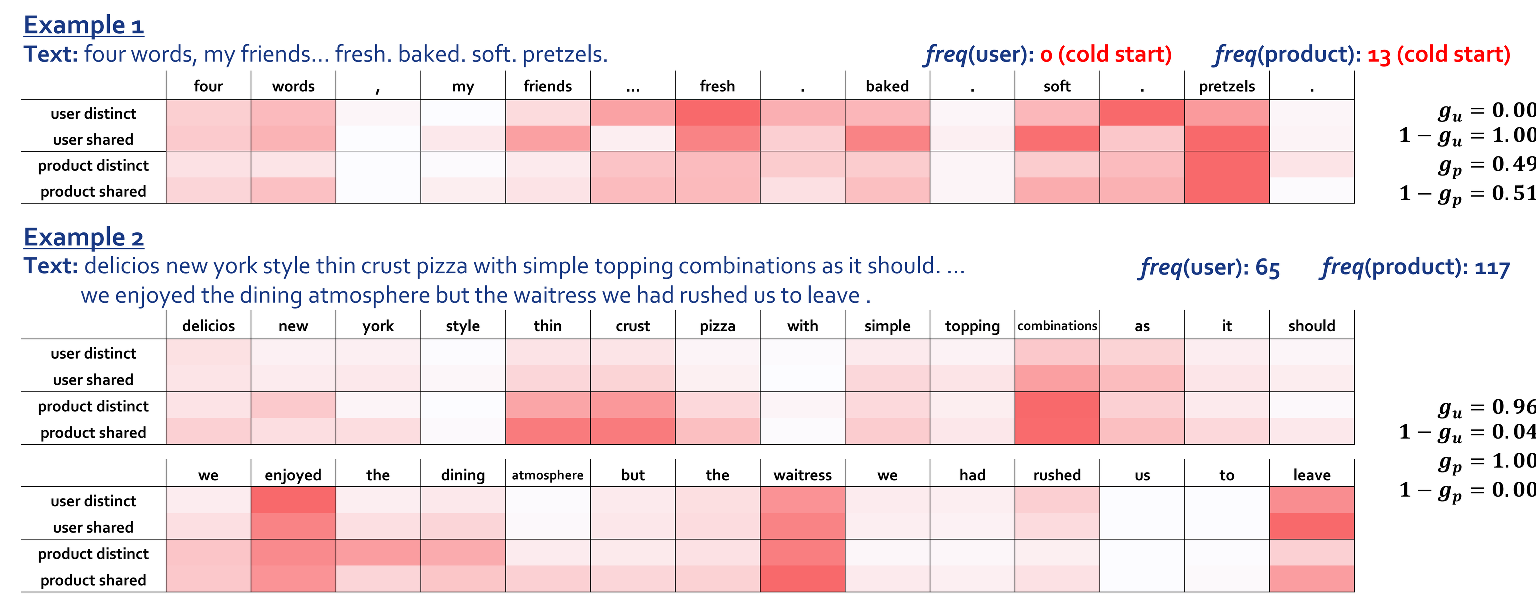
Q3: How few is **cold start**?



- Answer: we don't know!
- But HCSC provides insights through its **Weibull distribution**:

1. Users have a more lenient cold-start cut-off point than products.
2. It depends on the domain of the dataset (Yelp vs. IMDB).

Q4: How are **shared pooled vectors** made?



- They **create better representation** when users/products are cold-start (e.g. Example 1)
- They **imitate** the distinct pooled vectors when users/products are not cold-start (e.g. Example 2)

Acknowledgments

This work was supported by Microsoft Research Asia and the ICT R&D program of MSIT/IITP. [2017-0-01778, Development of Explainable Human-level Deep Machine Learning Inference Framework]