# TextGCN Model Experiment:

## 1.Experiment setting:

#Reviews text :7192

#Needs text: 1074

#Nodes in the graph: 12535 (contain words, review texts’ id and need texts’ id)

* Build the review text and needs text in the same graph.
* Training using 90% of review data (6473), randomly select 10% of review data (719) as validation set. Test on all of needs text (1074).
* No finetuning in this experiment since training and validating are all on the review data.
* CPU labels use the newly relabeled version from 0-9.
* Only frequency>5 words were chosen to build the graph (consistent with original paper)

## 2. Comparison with multitask model:

Comparing with multitask model (using relabeled CPU):

The arrow -> indicate: *TextGCN experiments -> Multitask experiments with finetuning* (analyzing using average embedding)

|  |  |
| --- | --- |
| CPU | (R@1 : 0.30->0.33 R@5: 0.79->0.81 ) |
| RAM | (R@1 : 0.63->0.65 R@5: 0.98-> 0.99 ) |
| HD | (R@1 : 0.41->0.46 R@5: 0.93->0.94 ) |
| GPU | (R@1 : 0.57->0.59 R@5: 0.94->0.99 ) |
| Screen | (R@1 : 0.68->0.73 R@5: 0.94->0.99 ) |

Although TextGCN performs not as good as Multitask results, but we can see that TextGCN achieves comparably good results since we don’t have explicit finetuning process in this experiment (actually it’s hard to finetune on TextGCN).

## 3. Result Analysis

Why TextGCN can achieve such high result:

It utilizes the overall information of words and each review, needs text. Each words appeared in reviews and needs have a link to review/need node. And also words have links to words weighted by PMI. The needs text is generated based on some keywords in review text. We can imagine that such kind of information is totally captured by the TextGCN model (Reviews and needs which contain similar words can get a similar embedding during training).

## 4.Result Table

|  |
| --- |
| TextGCN |
| CPU (newly relabeled):  ncdg:  0.29981378026070765  0.49813780260707635  0.5903686883712466  0.6294748336226428  0.6575450003499589  precision:  0.29981378026070765  0.24906890130353818  0.21477343265052762  0.18063314711359404  0.15754189944134078  recall:  0.29981378026070765  0.49813780260707635  0.6443202979515829  0.7225325884543762  0.7877094972067039 |
| HD:  current exper\_param is : laptop\_hd  ncdg:  0.4106145251396648  0.6769087523277467  0.7667898066074788  0.801706007724797  0.818147105379368  precision:  0.4106145251396648  0.33845437616387336  0.27312228429546864  0.2222998137802607  0.18547486033519553  recall:  0.4106145251396648  0.6769087523277467  0.819366852886406  0.8891992551210428  0.9273743016759777 |
| GPU:  current exper\_param is : laptop\_gpu  ncdg:  0.574487895716946  0.8156424581005587  0.8585268826915412  0.8771488566207778  0.8879759209298856  precision:  0.574487895716946  0.40782122905027934  0.2945375543140906  0.23021415270018622  0.18919925512104283  recall:  0.574487895716946  0.8156424581005587  0.8836126629422719  0.9208566108007449  0.9459962756052142 |
| Screen  current exper\_param is : laptop\_screen  ncdg:  0.6824953445065177  0.8538175046554934  0.896114471375367  0.9147364453046036  0.9259645119955301  precision:  0.6824953445065177  0.4269087523277467  0.3069522036002483  0.23952513966480446  0.19683426443202978  recall:  0.6824953445065177  0.8538175046554934  0.9208566108007449  0.9581005586592178  0.984171322160149 |
| RAM  current exper\_param is : laptop\_ram  ncdg:  0.6303538175046555  0.8156424581005587  0.8867248605047907  0.9132611733539526  0.9148651828812278  precision:  0.6303538175046555  0.40782122905027934  0.3094351334574798  0.24534450651769088  0.19702048417132215  recall:  0.6303538175046555  0.8156424581005587  0.9283054003724395  0.9813780260707635  0.9851024208566108 |