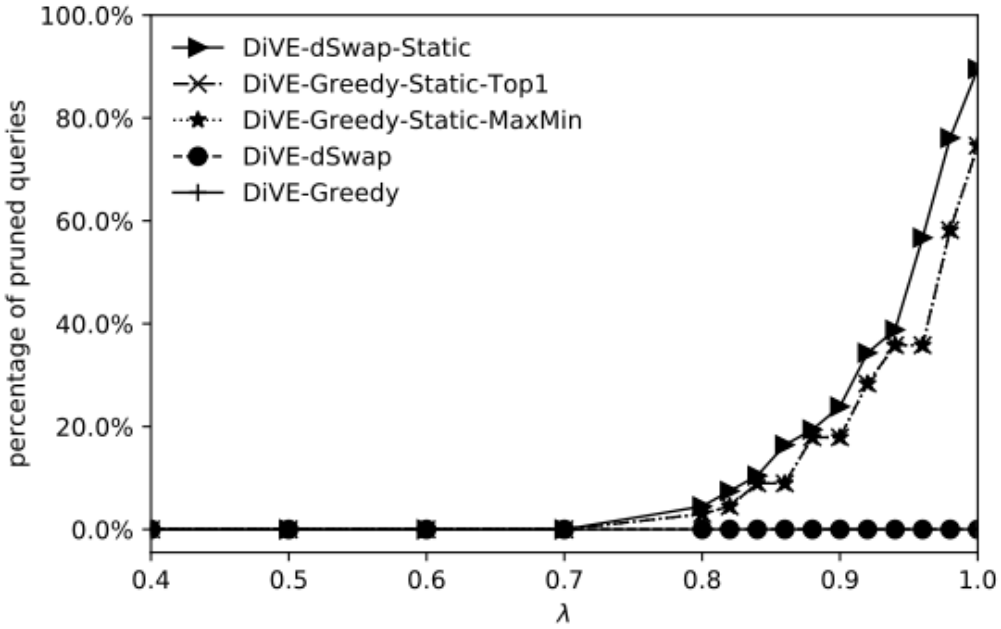
Greedy Static Max-Min vs Greedy Static Top-1

Greedy MaxMin and Greedy Top1 are very similar, both algorithms have same logics. First, as the initialization to get the set *S*, two most distant views are selected based on diversity. Then, all candidate views in *X* are sorted based on *setDist* (distance from view in *X* to set *S*).

In case of MaxMin pruning algorithm, *Umax* of all candidate views in *X* are computed using actual score of diversity and maximum bound of importance score while *Umin* of all candidate views in *X* are computed using actual score of diversity and 0 as the importance score. All views in *X* are pruned while the value of *Umax* less than the maximum of *Umin*. The view in *X* is executed one by one from the top, since the actual score of importance has been known, the actual utility score of the view can be calculated. This actual utility score of view will be used as the maximum of *Umin*. Since, more views are executed the higher actual utility will be used as the maximum of *Umin* and more views can be pruned.

Meanwhile, Top1 algorithm directly uses the total objective function of the set *F(S)*. After all candidate views in *X* are sorted based on *setDist*, the view is executed one by one start from the top. While the importance score of first executed view has been known, the $F(S \cup X[0])$ can be calculated and this objective function as the current objective function $F(S)$. Then, $maxF(S \cup X[i])$ of all remaining views in *X* are calculated using actual diversity score and maximum bound of importance score. If $maxF(S \cup X[i]) < F(S)$ then it is guaranteed that the actual objective function to be less than the current objective function $F(S)$ and those views can be pruned. The current objective function $F(S)$ is updated while the next importance score of executed query has higher score than the current.

Figure below shows that both algorithms have similar performance. This experiment using Flights dataset with 5 queries, I have checked that the result set *S* of MaxMin is equal to Top1.

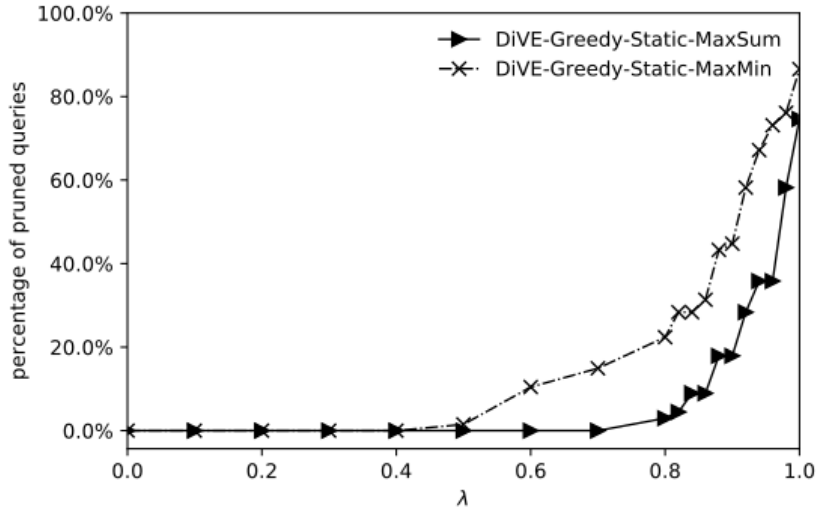


For the rest, we will use Top1 instead of MaxMin pruning due to we also have Swap technique that implement Top1 approach. The reason why we still need swap is because Greedy starts with small number of views in the initialization. Only two views as the set *S*. Hence, while calculating the *setDist* of each views in *X*, there are a lot of views have same score that it may decrease the chance of pruning. Moreover, there is no guarantee that two most distant views which selected by Greedy as the initialization have high importance score and there is no way to replace those two views. However, Swap has bigger number of views in the initialization (e.g. five views). While calculating the *setDist* of views in X, the chance of views have same score is lower than Greedy.

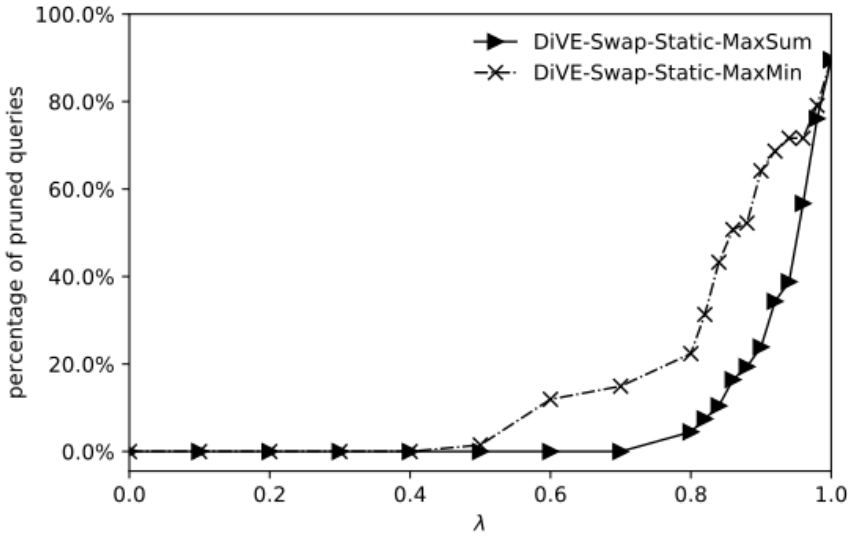
Additionally, swap also has a replacing mechanism that can replace low quality views in the current set with the better one.

Max-Sum Diversification vs. Max-Min Diversification

Current results that we have are using MaxSum diversification. Based on my hypothesis and experiments that MaxMin Diversification can improve the pruning performance. Below the results while using MaxSum vs. MaxMin on Greedy Static Top1.



Greedy Static Top1 using Max-Sum vs Max-Min diversification



dSwap Static Top1 using Max-Sum vs Max-Min diversification

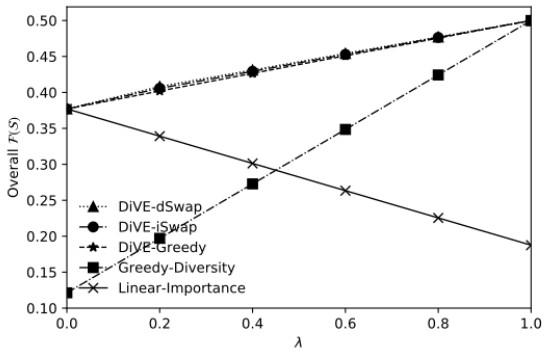
MaxSum uses average score of diversity of the set *S* which is by computing the total sum of all distances then dividing by *k*(k-1)* while MaxMin uses the maximum of minimum score of distance in the set *S*. Hence, the

range diversity score from those both approaches are different. For instance, there are three views in set Z which each view is different with others. The maximum score of distance between two views is 1 and the minimum is 0. Using MaxSum method the diversity score of set Z will be *(1+1+1)/(3*(3-1))* = 0.5 whereas diversity score of MaxMin is 1 because the minimum distance in the set Z is 1.

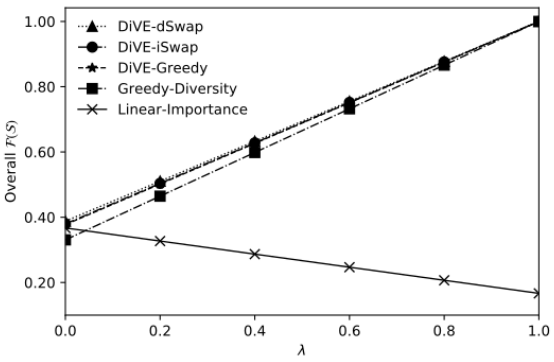
The example variance of *setDist* score using Flights dataset while λ =0.8 between MaxSum diversification and MaxMin diversification can be seen in the Table below. In this experiment, I selected two most distant views as the initial set S and then calculate the *setDist* of all views in *X*. For instance, the highest score of *setDist* is *v1*, where on MaxSum the maximum sore is 0.5 and on MaxMin the maximum score is 1. This Table is just an example, in the real data there are many views have same score. In this Table, I only want to show the distributions of *setDist* score and the different range of *setDist* score between MaxSum and MaxMin.

v in X	**MaxSum**	**MaxMin**
v1	0.5	1
v2	0.491666667	0.833333333
v3	0.483333333	0.666666667
v4	0.475	0.5
v5	0.466666667	0.333333333
v6	0.458333333	0.166666667
v7	0.45	0.166666667

Due to this different diversity score, MaxMin diversification can improve the pruning performance as shown in the Figure above. However, this MaxMin makes unbalance between the importance score and diversity score. The maximum diversity can be equal to 1 while the value of importance score is lower than that. This thing makes the shape of objective function unbalance. Please see Figures below for more details.



MaxSum diversification



MaxMin diversification

Datasets

There are several datasets that I tried in June, as follows: All datasets in Table blow and Airbnb dataset

No.	name	from
D1	*Happy Countries*	http://www.kenflerlage.com/2016/08/whats-happiest-country-in-world.html
D2	*US Baby Names*	https://deepsense.io/us-baby-names-data-visualization/
D3	*Flight Statistics*	https://www.transtats.bts.gov/airports.asp?pn=1
D4	*TutorialOfUCB*	https://multimedia.journalism.berkeley.edu/tutorials/data-visualization-basics/
D5	*CPI Statistics*	https://medium.com/towards-data-science/data-visualization
D6	*Healthcare*	https://getdataseed.com/demo/
D7	*Services Statistics*	https://getdataseed.com/demo/
D8	*PPI Statistics*	https://ppi.worldbank.org/visualization/ppi.html
D9	*Average Food Price*	http://data.stats.gov.cn/english/vchart.htm

From all datasets that I tried, still could not find the proper dataset for our work. I am still trying to find interesting dataset which has several numerical attributes and categorical attributes as well as big enough in terms of size.

Distance Functions

In terms of distance functions, I have been looking for the bounded distance function to compare between two probability distributions, there are several distance functions that I have studied:

1. Euclidean distance, currently is used and the maximum bound is $\sqrt{2}$. I sent the mathematically prove in the previous report.
2. Kullback-Leiber (KL) distance, this distance is not bounded, the mathematically proof has been sent in the previous report.
3. Earth Mover Distance (EMD), this distance is widely used and very good for comparing two probability distributions. Mostly this distance used in computer vision application to compare between two histograms. However, based on my finding, this distance is not bounded as well.
4. Kolmogorov Distance has the maximum bound equal to 1. However, mostly example code that I found uses this distance as hypothesis test that need another parameters such as α and confidence interval.

Until now, I am not sure if there is a bounded distance function that can be used for our work except the Euclidean distance.

Rectifying Bound on Adaptive Pruning Method

I am still working here, it's really complicated to store all results and make check point while the program running.