HW3- Paper Summary

Combining Hashing and Abstraction in Sparse High Dimensional Features Spaces

Today text data available online, e.g; news article, scientific documentation, weblogs etc, is increasing exponentially. So it requires effective and efficient classification methods to organize this data. The performance of the document classifiers critically depends on the choice of the features representation.

The “bag of words” and n-gram approaches construct a vocabulary of size d, which contains all words or n-grams in a collection of documents. A document is represented as a vector x with as many entries as the words or n-grams in the vocabulary. This approach verily fails because size of the vector would be very large and thus it will require O(d) space.

This paper talks about the techniques used to reduce the sparse high dimensional features spaces into low dimensional features spaces. Hashing and Abstraction are one of them. Applying clustering based on hashing and abstraction , we can reduce the dimensional features spaces to smaller number of features without loss of too much information.

Feature Hashing

In hashing, a high dimensional input vector x of size d is hashed into lower dimensional feature vector x^h of size b. Each token in the text document is directly mapped into a hash key. Also each index in the x^h stores the value(frequency count) of the corresponding hash features.

Thus an entry in the x^h records the frequency count of tokens which are hashed together randomly into the same hash key.

However, in a document collection, typically, only very few words occur with high frequency whereas most of them occur very rarely. Hence useful information necessary for high accuracy classification could be lost in feature hashing.

The value of b can be smaller than the theoretical lower bound. This would be problematic as the smaller the size of hash vector x^h becomes, the more the collisions occur in data. Therefore we can’t reduce dimensionality to great extent.

Feature Abstraction

Feature abstraction effectively reduces a classifier input size by clustering similar features into an abstraction hierarchy. It will reduce the space dimensions of a hash feature set from b to m (m<<b).

An abstraction hierarchy (AH) over H is defined as a rooted tree such that the leaf nodes correspond to the hash features in H and the internal nodes correspond to abstractions or clusters of hash features.

The algorithm initializes each abstraction with a hash feature in H, then recursively merges pairs of abstractions that are most “similar” to each other.

Combining Hashing and Abstraction

Since feature hashing can suffer from loss of information due to hashing in the same cluster of two high frequency features with significantly different class distributions, whereas feature abstraction can avoid such a pitfall by not allowing features in the same cluster, unless the class distributions of the two features are significantly similar.

Therefore we use both features by combining hashing and abstraction. It gives good results with less dimensional space vectors and less information loss.

References: Combining Hashing and Abstraction in Sparse High Dimensional Features Spaces

(Cornelia Caragea, Adrian Silvescu, Prasenjit Mitra)