FAMER: Making Multi-Instance Learning Better and Faster

Wei Ping*, Ye Xu§, Jianyong Wang*, Xian-Sheng Hua◊

* Tsinghua University
§ Dartmouth College
◊ Microsoft Research Asia
Outline

- Background
- Thinking in MIL (Multi-Instance Learning) Kernels
- Description of proposed FAMER (FAst kernel for Multi-instanceE leaRning)
- Experiment
- Conclusion
Multi-Instance Learning:

- Training example is bag containing instances.
- Bag label is known, but instance label is unseen.
- Bag is positive iff it contains at least one positive instance; otherwise it is negative.
Background: Multi-Instance Learning

Two perspectives for solving MIL

- Instance level
  - Eliminate the uncertainty of instance label in positive bag
  - Learn a model in instance space

- Bag level
  - View each bag as entity
  - Learn a model for bags.
Background: MIL Kernels

- MIL Kernel is bag level method
  - A powerful family of method in MIL
  - Kernel value between bags
  - Some typical MIL Kernels:
    - NSK and STK [Gartner, ICML02]
    - Marginalized Kernel [Kwok, IJCAI07]
    - PPMM Kernel [Wang, ICML08]
    - MI-Graph and mi-Graph [Zhou, ICML09]
    - ……
Outline

- Background
- **Thinking in MIL Kernels**
- Description of proposed FAMER
- Experiment
- Conclusion
Thinking in MIL Kernel

- Shortcomings of current MIL Kernels:
  - Correspondence information;
  - Co-occurrence information;
  - Assumption: all instances within a bag have equal weight (key instances)
  - Heavy computing load (pair-wise manner).
[Thinking in MIL Kernel: Correspondence]

○ The **correspondence** relationship between these two image bags (i.e., the respective coast, sea and sky segmentations/instances) indicates a high similarity.
Thinking in MIL Kernel: Correspondence

- NSK [Gartner, ICML02] gets decent results for most of real-world datasets

\[ K(X_i, X_j) = \frac{1}{n_in_j} \sum_{a=1}^{n_i} \sum_{b=1}^{n_j} k(x_{ia}, y_{jb}) \]

- Therein, \[ k(x_{ia}, y_{jb}) = \exp(-\gamma \| x_{ia} - y_{jb} \|^2) \]

- **Exponential amplification**: the relation between RBF Kernel Value and \textit{L2 distance using the empirical rule of thumb} indicated in [8]
Thinking in MIL Kernel: Correspondence

- Classification Accuracy of NSK using linear kernel vs. RBF kernel

<table>
<thead>
<tr>
<th>Data set</th>
<th>Musk1</th>
<th>Musk2</th>
<th>Elept</th>
<th>Fox</th>
<th>Tiger</th>
</tr>
</thead>
<tbody>
<tr>
<td>$NSK_{Linear}$</td>
<td>86.1%</td>
<td>82.4%</td>
<td>79.0%</td>
<td>55.5%</td>
<td>77.3%</td>
</tr>
<tr>
<td>$NSK_{RBF}$</td>
<td>88.0%</td>
<td>89.3%</td>
<td>84.3%</td>
<td>60.3%</td>
<td>84.2%</td>
</tr>
</tbody>
</table>

- The exponential amplification of RBF result in soft correspondence scheme: similar pair of instances make much more significant contribution for bag kernel
Thinking in MIL Kernel: Co-occurrence

- **Co-occurrence Information** (Non. I.I.D [Zhou, *ICML09*]):
  - In image bag, instances (segmentations) containing monkeys are very likely to co-occur with those instances containing trees.
Outline

- Background
- Thinking in MIL Kernel
- Description of proposed FAMER
- Experiment
- Conclusion
Locality Sensitive Hashing is a random distribution on a family $\mathcal{H}$ of hash functions, such that the probability of two objects being mapped to the same value reflects the similarity between them:

$$Pr_{h \in \mathcal{H}}[h(x) = h(y)] = s_{\mathcal{H}}(x, y)$$

Given an arbitrary locally sensitive hash function $h \in \mathcal{H}$, we define the corresponding induced similarity measure as:

$$s_h(x, y) = \delta_{h(x), h(y)}$$

Therein, $\delta$ is Kronecker delta. From (3.1) and (3.2),

$$E_{h \in \mathcal{H}}[s_h(x, y)] = s_{\mathcal{H}}(x, y)$$
FAMER: Construct Similarity Metric with Correspondence

- In order to capture correspondence
  - A LSH based similarity metric: by analogy from NSK,

\[
S_{MI,\mathcal{H}}(X_i, X_j) = \frac{1}{n_i n_j} \sum_{a=1}^{n_i} \sum_{b=1}^{n_j} \exp(-\gamma(1 - s_{\mathcal{H}}(x_{ia}, x_{jb})))
\]

- Therein, \( s_{\mathcal{H}}(x_{ia}, y_{jb}) \) is induced from LSH family \( \mathcal{H} \) as (3.1)
For each $h \in \mathcal{H}$, we define

\[ S_{MI,h}(X_i, X_j) = \frac{1}{n_in_j} \sum_{a=1}^{n_i} \sum_{b=1}^{n_j} \exp(-\gamma(1 - s_h(x_{ia}, y_{jb}))) \]

We could use $E_{h \in \mathcal{H}}[S_{MI,h}(X_i, X_j)]$ to approximate $S_{MI,\mathcal{H}}(X_i, X_j)$

Lemma: (3.4) $S_{MI,\mathcal{H}}(X_i, X_j)$ is the lower bound of $E_{h \in \mathcal{H}}[S_{MI,h}(X_i, X_j)]$
Pair-wise manner is still used in (3.5): time consuming.

Use embedding histogram to reduce the time complexity: \( T_h(X_i) = \sum_{x_{ij} \in X_i} e[h(x_{ij})] \)

No longer compute in pair-wise manner

\( (3.5) \)

\[
S_{MI,h}(X_i, X_j) = \frac{1}{n_in_j} \sum_{a=1}^{n_i} \sum_{a=1}^{n_j} exp(-\gamma(1 - s_h(x_{ia}, y_{jb})))
\]

\[
= \frac{1}{n_in_j} \left\{ T_h(X_i) \cdot T_h(X_j) + [n_in_j - T_h(X_i) \cdot T_h(X_j)] \times exp(-\gamma) \right\}
\]
FAMER: Weighting Scheme

- Imposing **high weights** on “key” instances

**Motivation:**
- Instances mapped in the **same bin**; similar instances under some specific probabilities.
- Each **bin** in the histogram corresponds to a probabilistic **region** in instance space.
- As DD pointed, regions with **many negative** and **few “positive” instance**, have much discriminative information for **negative bags**, and vice versa.
FAMER: Weighting Scheme

- \( \Delta \text{Entropy} \) is employed to evaluate the “purity” of bin(region), i.e. the weight of each bin[i].

\[
P_+[i] = \frac{N_+[i] + \epsilon}{N_+[i] + N_-[i] + 2\epsilon} \quad P_-[i] = \frac{N_-[i] + \epsilon}{N_+[i] + N_-[i] + 2\epsilon}
\]

\[
W[i] = \Delta H[i] + C = H_0[i] - H_e[i] + C
\]

\[
= \ln 2 - (P_+[i] \ln \frac{1}{P_+[i]} + P_-[i] \ln \frac{1}{P_-[i]}) + C
\]

- In this way, instances within each bin are implicitly weighted.
FAMER: Weighting Scheme

- False Positive problem:
  - In practice, we regard all instances in positive bags as “positive” when estimating the probability.
  - Influence of those false positive(negative) instances in positive bags, can be offset by the negative instances from corresponding negative regions indicated by negative bags.
FAMER: Weighting Scheme captures Co-occurrence

- Weighting Scheme also detect the co-occurrence information
  - The histogram naturally records the co-occurrence information of "positive"/negative instances.
  - The joint statistics respects the co-occurrence relations.
Final form of FAMER:

\[(3.6)\]
\[K_{MI, HM}(X_i, X_j) = \frac{1}{M} \frac{1}{n_i n_j} \left\{ (W \circ T_{HM}(X_i)) \cdot T_{HM}(X_j) + [M n_i n_j \right.\]
\[-(W \circ T_{HM}(X_i)) \cdot T_{HM}(X_j)] * \exp(-\gamma) \left. \right\} \]

- Therein, \( M \) is the number of sampling
- **Lemma:** \( K_{MI, HM}(X_i, X_j) \) is a Mercer Kernel
FAMER: Time Complexity

- Embedding + Computing of Kernel Matrix
- $O(C_1 \cdot dnN + C_2 \cdot N^2)$
  - \(N\): number of bags;
  - \(n\): average instances number within each bag;
  - \(d\): dimensionality of feature vector.
  - \(C_1\) and \(C_2\) are constant depend on parameters
- Linear with \(n\): efficient, compared with other typical MIL Kernels (Quadratic with \(n\)).
Outline

- Background
- Thinking in MIL Kernel
- Description of FAMER
- Experiment
- Conclusion
Experiments

- Effectiveness vs. Efficiency
- Three task:
  - Drug Activity Prediction
    - Musk1, Musk2
  - Automatic Image Annotation
    - Elephant, Fox, Tiger
  - Identifying Trx-fold Proteins
    - Protein
Experiments

- Comparison
  - Typical MIL Kernels: NSK, STK, MG-ACC Kernel, PPMM Kernel, mi-Graph Kernel
  - Other famous MIL algorithms: mi-SVM, MI-SVM, EM-DD
- Use 10 times 10 CV to obtain Classification Accuracy
Experiments: Effectiveness (1)

Classification Accuracy on Musk data set

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Musk1</th>
<th>Musk2</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>$NSK_{rbf}$ [8]</td>
<td>88.0%</td>
<td>89.3%</td>
<td>88.7%</td>
</tr>
<tr>
<td>STK [8]</td>
<td>91.6%</td>
<td>86.3%</td>
<td>89.0%</td>
</tr>
<tr>
<td>MIGraph Kernel [13]</td>
<td>90.0%</td>
<td>90.0%</td>
<td>90.0%</td>
</tr>
<tr>
<td>miGraph Kernel [13]</td>
<td>88.9%</td>
<td>90.3%</td>
<td>89.6%</td>
</tr>
<tr>
<td>MG-ACC Kernel [12]</td>
<td>90.1%</td>
<td>90.4%</td>
<td>90.3%</td>
</tr>
<tr>
<td>PPMM Kernel [21]</td>
<td>95.6%</td>
<td>81.2%</td>
<td>88.4%</td>
</tr>
<tr>
<td><strong>FAMER</strong></td>
<td>91.3%</td>
<td>93.3%</td>
<td>92.3%</td>
</tr>
<tr>
<td>MI-SVM [9]</td>
<td>77.9%</td>
<td>84.3%</td>
<td>81.1%</td>
</tr>
<tr>
<td>mi-SVM [9]</td>
<td>87.4%</td>
<td>83.6%</td>
<td>85.5%</td>
</tr>
<tr>
<td>MissSVM [33]</td>
<td>87.6%</td>
<td>80.0%</td>
<td>83.8%</td>
</tr>
<tr>
<td>DD [6]</td>
<td>88.0%</td>
<td>84.0%</td>
<td>86.0%</td>
</tr>
<tr>
<td>EM-DD [34]</td>
<td>84.8%</td>
<td>84.9%</td>
<td>84.9%</td>
</tr>
<tr>
<td>APR [1]</td>
<td>92.4%</td>
<td>89.2%</td>
<td>90.8%</td>
</tr>
<tr>
<td>MI-Box [30]</td>
<td>91.2%</td>
<td>90.3%</td>
<td>90.8%</td>
</tr>
</tbody>
</table>
Experiments: Effectiveness (2)

Classification Accuracy on Automatic Image Annotation

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Elephant</th>
<th>Fox</th>
<th>Tiger</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N S K_{rbf}$</td>
<td>84.3%</td>
<td>60.3%</td>
<td>84.2%</td>
</tr>
<tr>
<td>STK</td>
<td>83.5%</td>
<td>63.0%</td>
<td>79.0%</td>
</tr>
<tr>
<td>MIGraph Kernel</td>
<td>85.1%</td>
<td>61.2%</td>
<td>81.9%</td>
</tr>
<tr>
<td>miGraph Kernel</td>
<td>86.8%</td>
<td>61.6%</td>
<td>86.0%</td>
</tr>
<tr>
<td>PPMM Kernel</td>
<td>82.4%</td>
<td>60.3%</td>
<td>80.2%</td>
</tr>
<tr>
<td><strong>FAMER</strong></td>
<td><strong>87.5%</strong></td>
<td><strong>67.0%</strong></td>
<td><strong>87.0%</strong></td>
</tr>
<tr>
<td>MI-SVM</td>
<td>81.4%</td>
<td>59.4%</td>
<td>84.0%</td>
</tr>
<tr>
<td>mi-SVM</td>
<td>82.0%</td>
<td>58.2%</td>
<td>78.9%</td>
</tr>
<tr>
<td>EM-DD</td>
<td>78.3%</td>
<td>56.1%</td>
<td>72.1%</td>
</tr>
</tbody>
</table>
True Positive (TP) and True Negative (TN) typical Kernel under TrX Protein.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>TP</th>
<th>TN</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>$NSSK_{rbf}$</td>
<td>69.0%</td>
<td>72.6%</td>
<td>70.8%</td>
</tr>
<tr>
<td>STK</td>
<td>58.1%</td>
<td>66.1%</td>
<td>62.1%</td>
</tr>
<tr>
<td>MIGraph Kernel</td>
<td>68.1%</td>
<td>70.3%</td>
<td>69.2%</td>
</tr>
<tr>
<td>miGraph Kernel</td>
<td>64.1%</td>
<td>69.7%</td>
<td>66.9%</td>
</tr>
<tr>
<td>FAMER</td>
<td>70.1%</td>
<td>70.9%</td>
<td>70.5%</td>
</tr>
</tbody>
</table>
Experiments: Efficiency

- Running Time and Detailed Description of Dataset

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Musk1</th>
<th>Musk2</th>
<th>Elephant</th>
<th>Fox</th>
<th>Tiger</th>
<th>Protein</th>
</tr>
</thead>
<tbody>
<tr>
<td>$NSK_{rbf}$</td>
<td>375ms</td>
<td>67,562ms</td>
<td>4,438ms</td>
<td>4,015ms</td>
<td>3,453ms</td>
<td>161,844ms</td>
</tr>
<tr>
<td>MIGraph Kernel</td>
<td>1218ms</td>
<td>784,422ms</td>
<td>9,875ms</td>
<td>9,203ms</td>
<td>7,816ms</td>
<td>7,123,830ms</td>
</tr>
<tr>
<td>miGraph Kernel</td>
<td>385ms</td>
<td>77,485ms</td>
<td>5,250ms</td>
<td>4,703ms</td>
<td>4,016ms</td>
<td>199,461ms</td>
</tr>
<tr>
<td>FAMER</td>
<td>609 + 169</td>
<td>9,431 + 461</td>
<td>1704 + 500</td>
<td>1625 + 501</td>
<td>1502 + 498</td>
<td>4,359 + 3,875</td>
</tr>
<tr>
<td></td>
<td>= 778ms</td>
<td>= 9,892ms</td>
<td>= 2,204ms</td>
<td>= 2,126ms</td>
<td>= 2,000ms</td>
<td>= 8,234ms</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Musk1</th>
<th>Musk2</th>
<th>Elephant</th>
<th>Fox</th>
<th>Tiger</th>
<th>Protein</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of bags</td>
<td>92</td>
<td>102</td>
<td>200</td>
<td>200</td>
<td>200</td>
<td>180</td>
</tr>
<tr>
<td>Number of positive bags</td>
<td>47</td>
<td>39</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>20</td>
</tr>
<tr>
<td>Number of negative bags</td>
<td>45</td>
<td>63</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>160</td>
</tr>
<tr>
<td>Average number of instances per bag</td>
<td>5.2</td>
<td>64.7</td>
<td>6.96</td>
<td>6.6</td>
<td>6.1</td>
<td>137.8</td>
</tr>
<tr>
<td>Dimensionality of instance space</td>
<td>166</td>
<td>166</td>
<td>230</td>
<td>230</td>
<td>230</td>
<td>8</td>
</tr>
</tbody>
</table>
Experiments: Efficiency

- Three observation:
  - FAMER: best efficiency on 5/6 dataset
  - Under small data set, FAMER is averagely better than typical Multi-Instance Kernels
  - As the number of instances increases in every bag, the superiority of FAMER becomes more and more significant
Outline

- Background
- Thinking in MIL Kernel
- Description of FAMER
- Experiment
- Conclusion
FAMER: Fast kernel for MIL

- Captures valuable correspondence and co-occurrence information.
- Avoid pair-wise manner to speed up the computation.
- Captures key instances in MIL by Weighting scheme.
Thanks!