Доклад о работах Pedro Domingos из Вашингтонского университета

Спецсеминар "Алгебра над алгоритмами и эвристиеский поиск закономерностей"

Евгений Никишин

ВМК МГУ

14.10.2015

Краткие сведения

- Профессор в Вашингтонском Университете
- Автор и соавтор более 200 публикаций, 27000 цитирований
- Соучредитель International Machine Learning Society
- SIGKDD Innovation Award
- NSF CAREER Award
- Fulbright Scholarship
- IBM Faculty Award
- Several best paper awards

Pedro Domingos

1. Краткие сведения

2. Mining High-Speed Data Streams

3. Sum-Product Networks

- Hoeffding trees — могут быть обучены за константное время на объект.
- Вероятность того, что обычное дерево и Hoeffding tree выберут разные ответы, падает экспоненциально с количеством экземпляров.

- Very Fast Decision Tree learner (VFDT) — обрабатывает объекты за время, меньшее, чем требуется на их извлечение с диска.
- При этом объекты просматриваются лишь раз и никуда не записываются, поэтому необходима память только на место, пропорциональное размеру дерева.

Классические модели (вроде C4.5, CART) предполагают, что вся тренировочная выборка может быть записана в памяти единовременно, поэтому появляются ограничения. В Hoeffding trees делается предположение, что данных потенциально бесконечное число. Только первые экземпляры требуются для определения предиката в корневой вершине.

Проблема определения необходимого в каждой вершине количества экземпляров решается с помощью Hoeffding-Chernoff bound.
Пусть r — вещественная случайная величина, целиком расположенная на отрезке радиуса R. Пусть n — число независимых наблюдений и $\bar{r}$ — их среднее.
Hoeffding-Chernoff bound: с вероятностью $1 - \delta$ настоящее среднее величины не меньше $\bar{r} - \epsilon$, где

$$\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}}$$

Не зависим от распределения, но потребуется больше наблюдений.

Пусть $G(X_i)$ — мера информативности, используемая для определения тестового признака (как индекс Джини в CART). Задача: удостовериться, что, с высокой вероятностью, выбранный на основании n экземпляров (где n как можно меньше) признак будет таким же, как и выбранный на основании бесконечной выборки.

Пусть G максимизируется, X_a — признак с максимальным $\overline{G}$, X_b — второй лучший.
Положим $\Delta\overline{G} = \overline{G}(X_a) - \overline{G}(X_b) \geqslant 0$. При желаемом δ Hoeffding-Chernoff bound гарантирует, что X_a — верный выбор с вероятностью $1 - \delta$ при условии n наблюдений и $\Delta\overline{G} > \epsilon$ (т.е. $\Delta G \geqslant \Delta\overline{G} - \epsilon > 0$ с вероятностью $1 - \delta$)
Узлу необходимо накапливать объекты до тех пор, пока ϵ не станет меньше $\Delta\overline{G}$ (так как ϵ — монотонно убывающая функция от n). После этого узел можно разбить.

Procedure HoeffdingTree $(S, \mathbf{X}, G, \delta)$
Let HT be a tree with a single leaf l_1 (the root).
Let $\mathbf{X_1} = \mathbf{X} \cup \{X_\emptyset\}$.
Let $\overline{G}_1(X_\emptyset)$ be the $\overline{G}$ obtained by predicting the most frequent class in S.
For each class y_k
  For each value x_{ij} of each attribute $X_i \in \mathbf{X}$
    Let $n_{ijk}(l_1) = 0$.
For each example $(\mathbf{x}, y_k)$ in S
  Sort $(\mathbf{x}, y)$ into a leaf l using HT.
  For each x_{ij} in $\mathbf{x}$ such that $X_i \in \mathbf{X}_l$
    Increment $n_{ijk}(l)$.
  Label l with the majority class among the examples seen so far at l.
  If the examples seen so far at l are not all of the same class, then
    Compute $\overline{G}_l(X_i)$ for each attribute $X_i \in \mathbf{X}_l - \{X_\emptyset\}$ using the counts $n_{ijk}(l)$.
    Let X_a be the attribute with highest $\overline{G}_l$.
    Let X_b be the attribute with second-highest $\overline{G}_l$.
    Compute ϵ using Equation 1.
    If $\overline{G}_l(X_a) - \overline{G}_l(X_b) > \epsilon$ and $X_a \neq X_\emptyset$, then
      Replace l by an internal node that splits on X_a.
      For each branch of the split
        Add a new leaf l_m, and let $\mathbf{X_m} = \mathbf{X} - \{X_a\}$.
        Let $\overline{G}_m(X_\emptyset)$ be the $\overline{G}$ obtained by predicting the most frequent class at l_m.
        For each class y_k and each value x_{ij} of each attribute $X_i \in \mathbf{X_m} - \{X_\emptyset\}$
          Let $n_{ijk}(l_m) = 0$.
Return HT.

Пояснения к алгоритму

- Разбиение будет сделано только в случае, если, с вероятностью $1 - \delta$, согласно G лучше разбить, чем не разбивать.
- Выборка может быть бесконечной.
- Сложность по памяти не зависит от размера выборки.
- Получаемое дерево асимптотически близко к деревьям, использующим всю выборку для определения наилучшего подразбиения

- $\Delta_e(DT_1, DT_2) = \sum_{\mathbf{x}} P(\mathbf{x}) I[DT_1(\mathbf{x}) \neq DT_2(\mathbf{x})]$
- $\Delta_i(DT_1, DT_2) = \sum_{\mathbf{x}} P(\mathbf{x}) I[Path_1(\mathbf{x}) \neq Path_2(\mathbf{x})]$
- $\Delta_i(DT_1, DT_2) \geqslant \Delta_e(DT_1, DT_2)$
- p_l — вероятность объекта попасть в лист, дойдя до уровня l. Допущение: $\forall_l p_l = p$
- $E[\Delta_i(HT_\delta, DT_*)]$ — ожидаемое значение, взятое по всем возможным бесконечным выборкам.

Теорема: если HT_δ есть дерево, полученное данным алгоритмом с желаемой вероятностью δ, DT_* есть асимптотическое обыкновенное дерево, то $\mathrm{E}[\Delta_\mathrm{i}(\mathrm{HT}_\delta, \mathrm{DT}_*)] \leqslant \delta/\mathrm{p}$

Следствия:

- Δ_e тоже меньше (причем гораздо) δ/p
- Существует поддерево асимптотического обычного дерева такое, что Δ_e меньше δ/p

Пользователь может задать приемлемые для себя значения. К примеру, если разница между лучшим и вторым признаком 10% (т.е. $\epsilon/\mathrm{R} = 0.1$), для получения $\delta = 0.1\%$ необходимо 380 объектов, а для улучшения $\delta = 0.0001\%$ необходимо всего лишь 345 дополнительных объектов.

Свойства:

- Если два признака очень похожи (согласно выбранному G), VFDT сам выбирает необходимый признак.
- Неэффективно пересчитывать G, поэтому VFDT позволяет установить пользователю n_{min}, после которого будет пересчитан G.
- Если VFDT разрастается до таких размеров, когда перестает помещаться в оперативную память, наименее перспективные узлы выбрасываются.
- Система может быть инициализирована обычным деревом, обученным на подвыборке.
- VFDT может просматривать заново прошлые объекты, если новые поступают медленно.

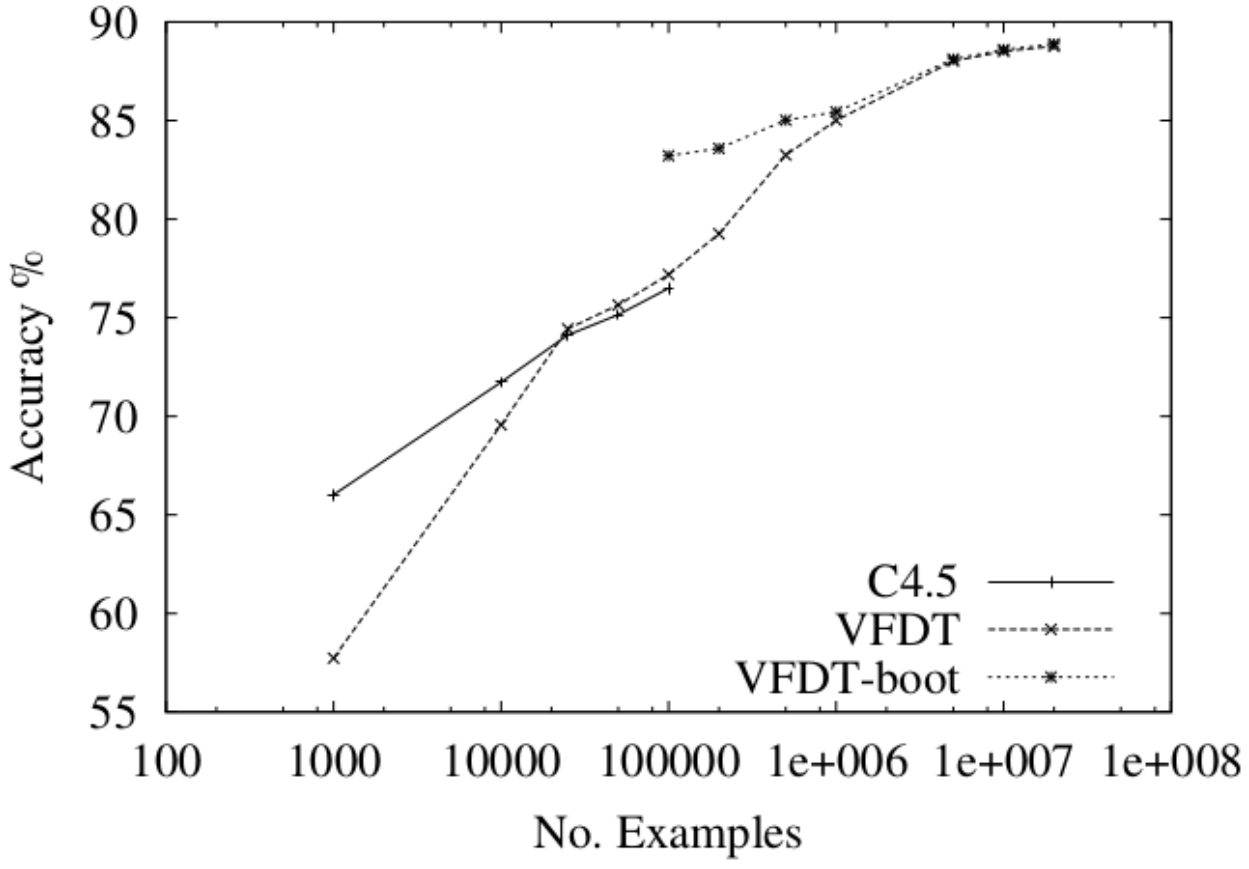


Figure 1: Accuracy as a function of the number of training examples.

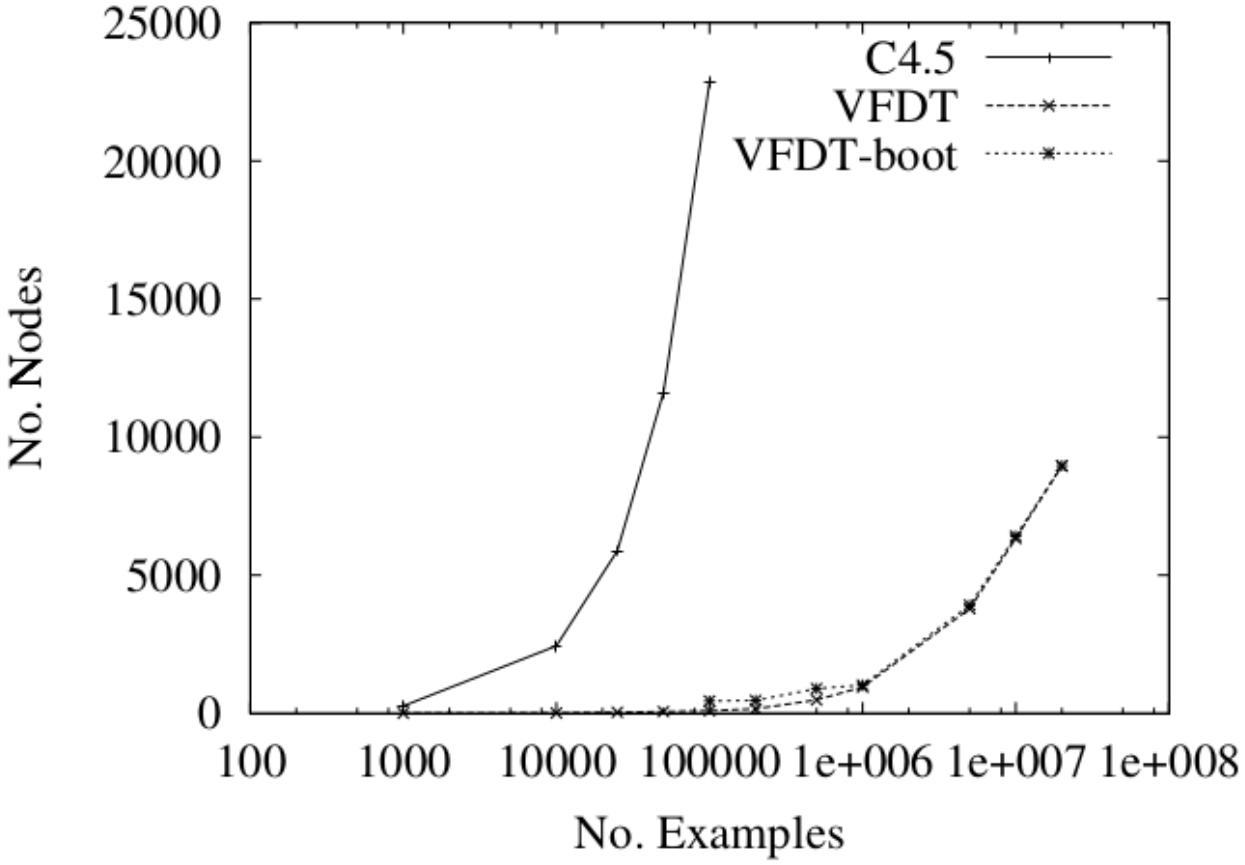


Figure 2: Tree size as a function of the number of training examples.

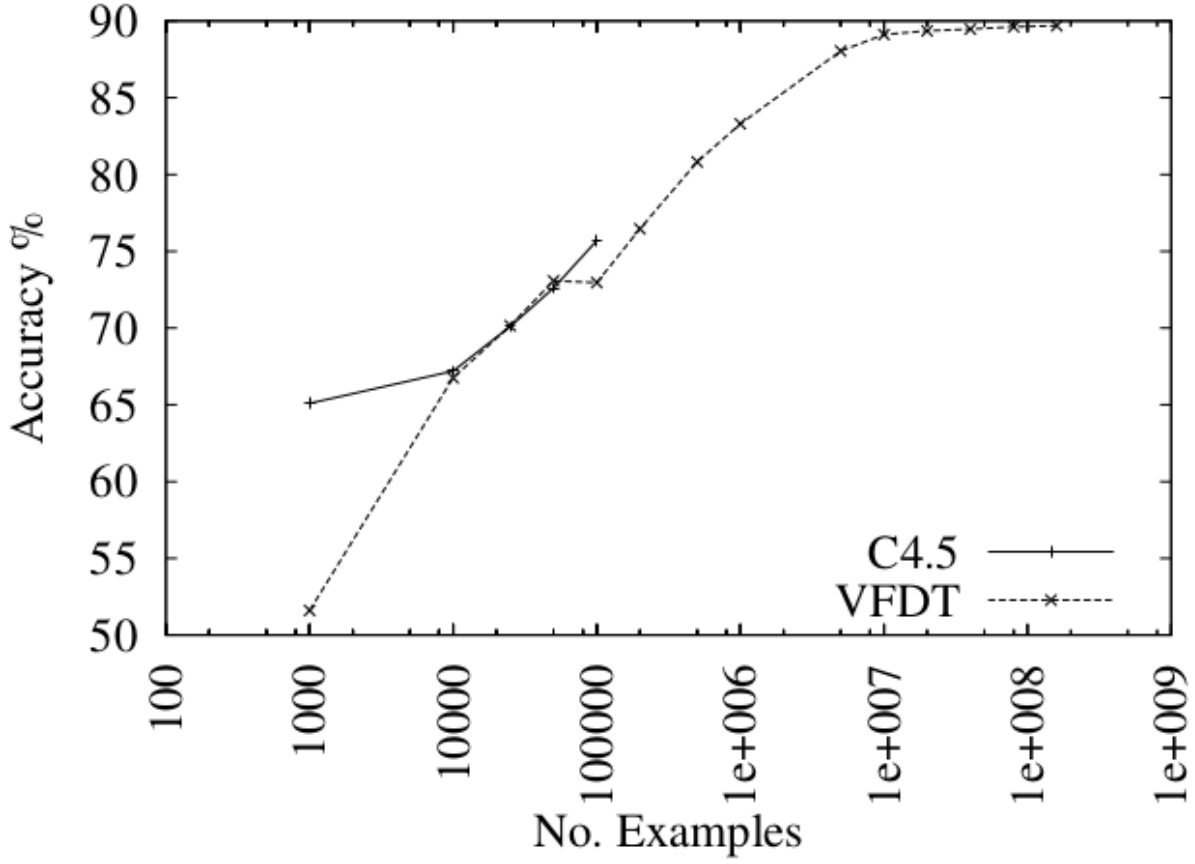


Figure 5: VFDT trained on 160 million examples.

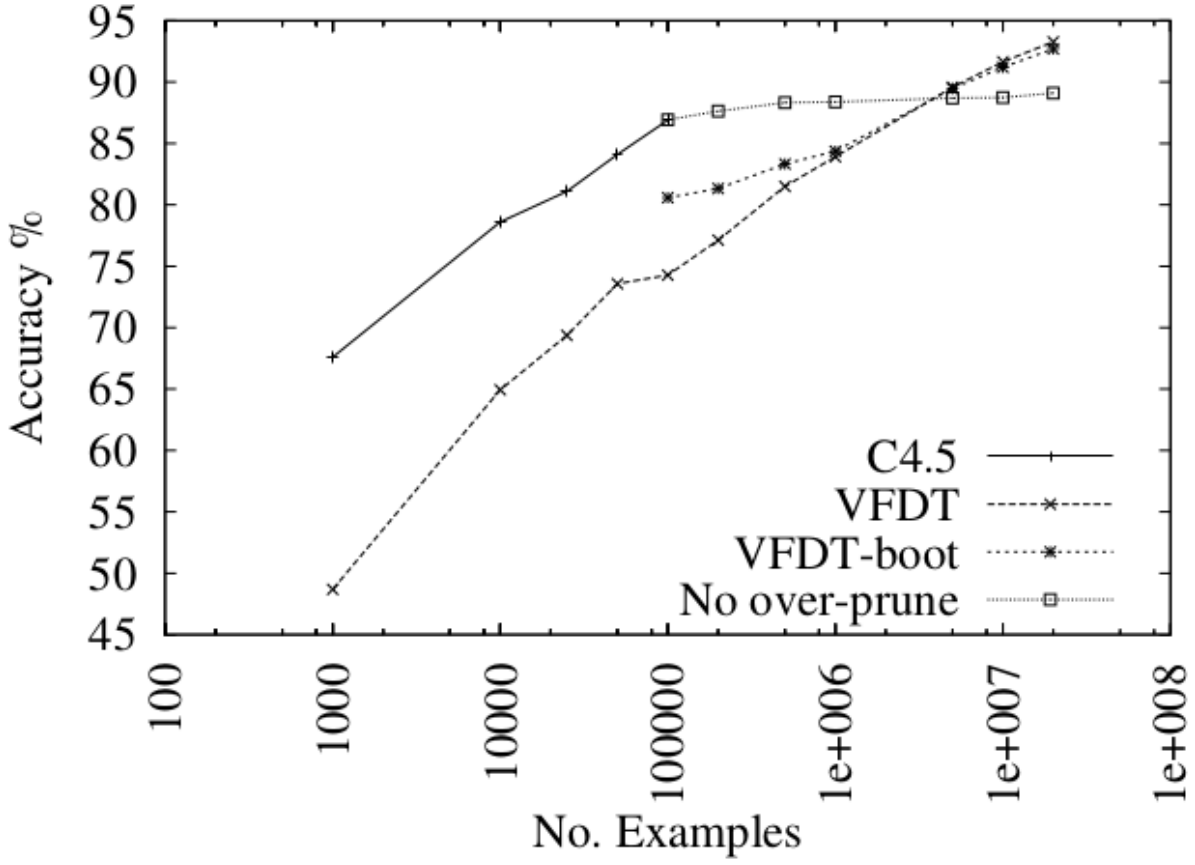


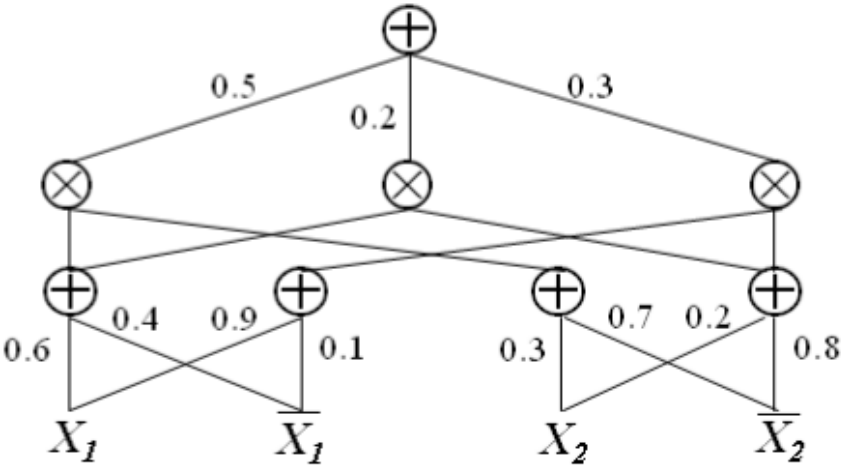
Figure 6: Effect of initializing VFDT with C4.5 with and without over-pruning.

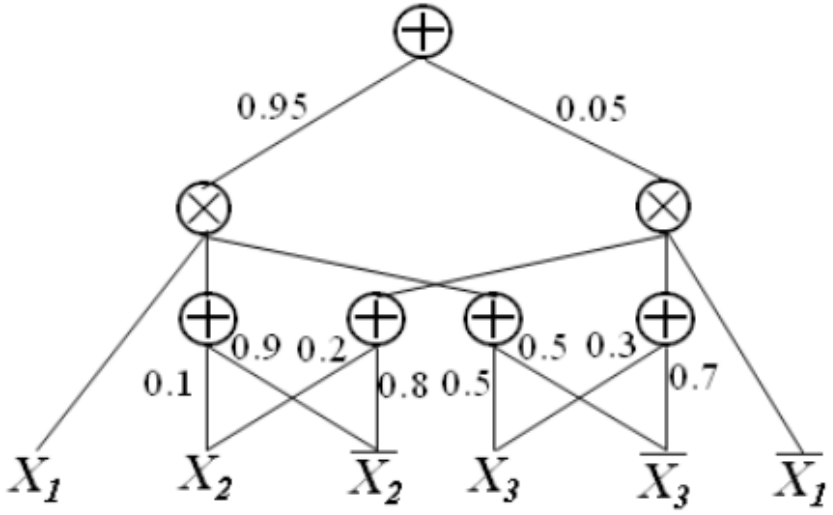
1. Краткие сведения

2. Mining High-Speed Data Streams

3. Sum-Product Networks

SPN над переменными $x_1, \ldots, x_d$ — направленный ациклический граф, имеющий в качестве листьев индикаторы $x_1, \ldots, x_d$ и $\overline{x}_1, \ldots, \overline{x}_d$, а в качестве внутренних узлов суммы и произведения. Каждое ребро (i, j), исходящее из узла i с суммой, имеет неотрицательный вес w_{ij}. Значением узла с произведением является произведение значений потомков. Значением узла с суммой является $\sum_{j \in Ch(i)} w_{ij} v_j$, где $Ch(i)$ — потомки i, v_j — значение узла j. Значением SPN является значение корня.


0.5
0.2
0.3
0.4
0.9
0.7
0.2
0.6
0.1
0.3
0.8
X_1
$\overline{X_1}$
X_2
$\overline{X_2}$


0.95
0.05
0.9
0.2
0.5
0.3
0.1
0.8
0.5
0.7
X_1
X_2
$\overline{X_2}$
X_3
$\overline{X_3}$
$\overline{X_1}$

Предположения:

- Граф "хороший"
- Слои из сумм и произведений чередуются

Algorithm 1 LearnSPN

Input: Set D of instances over variables X.
Output: An SPN with learned structure and parameters.
$S \leftarrow$ GenerateDenseSPN(X)
InitializeWeights(S)
repeat
  for all $d \in D$ **do**
    UpdateWeights(S, Inference(S, d))
  end for
until convergence
$S \leftarrow$ PruneZeroWeights(S)
return S

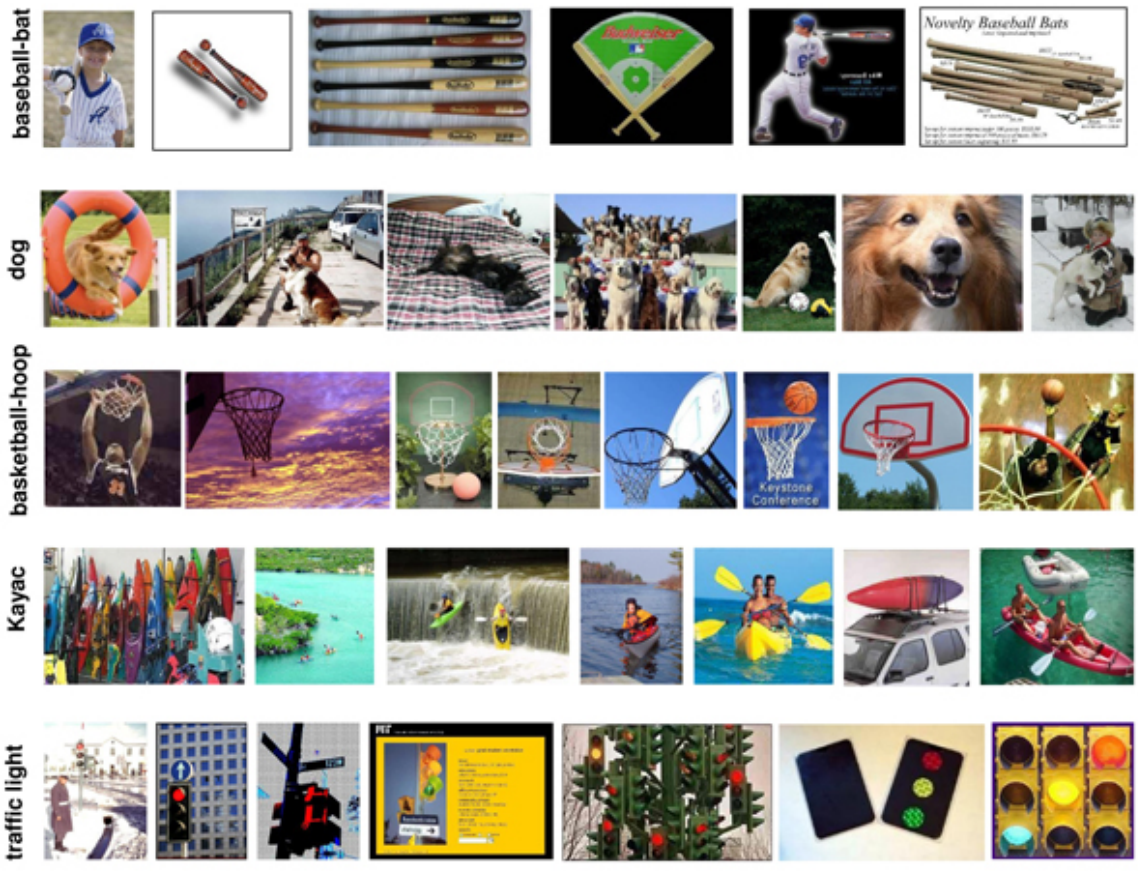
1. Выбираем множество подмножеств переменных.
2. Для каждого подмножества R, создаем k узлов с суммой $S_1^R, \ldots, S_k^R$ и выбираем множество путей для разложения R в другие -подмножества $R_1, \ldots, R_l$.
3. Для каждого из этих разложений и для всех $1 \leqslant i_1, \ldots, i_l \leqslant k$ создаем узел с произведением с родителями S_j^R и потомками $S_{i_1}^{R_1}, \ldots, S_{i_l}^{R_l}$.

Для многих областей интуитивно понятны выборы подмножеств и разложений. В противном случае можно выбирать случайно, как в случайных лесах.

Обновление весов производится либо с помощью градиентного спуска, либо с помощью EM. Также позже необходимо обрезать ребра с нулевыми весами.

Sum-Product Networks

Использование



- 36 слоев
- Все веса инициализированы нулями
- Все подразбиения — множество всех прямоугольников.
- Сравнение с Deep Belief Networks, Deep Boltzmann machines, Principal Component Analysis, kNN

Table 1: Mean squared errors on completed image pixels in the left or bottom half. NN is nearest neighbor.

LEFT	SPN	DBM	DBN	PCA	NN
Caltech (ALL)	3551	9043	4778	4234	4887
Face	1992	2998	4960	2851	2327
Helicopter	3284	5935	3353	4056	4265
Dolphin	2983	6898	4757	4232	4227
Olivetti	942	1866	2386	1076	1527

BOTTOM	SPN	DBM	DBN	PCA	NN
Caltech (ALL)	3270	9792	4492	4465	5505
Face	1828	2656	3447	1944	2575
Helicopter	2801	7325	4389	4432	7156
Dolphin	2300	7433	4514	4707	4673
Olivetti	918	2401	1931	1265	1793

Figure 5: Sample face completions. Top to bottom: original, SPN, DBM, DBN, PCA, nearest neighbor. The first three images are from Caltech-101, the rest from Olivetti.