**TASK2VEC**: Task Embedding for Model Recommendation

**Subhransu Maji**  
College of Information and Computer Sciences  
University of Massachusetts, Amherst  
http://people.cs.umass.edu/smaji

February 19, 2019 @ ICERM, Brown University

Task Embedding for Model Recommendation

Allesandro, Michael, Rahul, Avinash, Subhransu, Charless, Stefano, Pietro

Task = \{\text{dataset, labels, loss}\}

What are similar tasks?
What architecture should I use?
What pre-training dataset?
What hyper parameters?
Do I need more training data?
How difficult is this task?

If we have a universal vectorial representation of tasks we can frame all sorts of interesting CV applications engineering problems as meta-learning problems.
Model recommendation

**Brute Force:**

**Input:** Task = (dataset, loss)

**For each** feature extractor architecture $F$:
1. Train **classifier** on $F(dataset)$
2. Compute validation performance

**Output:** best performing model

**Task recommendation:**

**Input:** Task = (dataset, loss)

1. Compute task embedding $t = E(Task)$
2. Predict best extractor $F = M(t)$
2. Train **classifier** on $F(dataset)$
3. Compute validation performance

**Output:** best performing model
Task embedding using Fisher Information

1. Given a **task**, train a classifier with the **task loss** on features from a generic “probe network”

2. Compute gradients of probe network parameters w.r.t. task loss

3. Use statistics of the probe parameter gradients as the fixed dimensional task embedding

\[
F = \frac{1}{N} \sum_{i=1}^{N} \nabla \log p(x_i|\theta) \nabla \log p(x_i|\theta)^T
\]

Intuition: F provides information about the sensitivity of the task performance to small perturbations of parameters in the probe network

\[
\mathbb{E}_{x \sim \hat{p}} KL_{p_{\theta'}}(y|x)p_{\theta}(y|x) = \delta \theta \cdot F \cdot \delta \theta + o(\delta \theta^2),
\]
Properties of Task2vec embedding

Dataset:

\[(x_i, y_i), i = 1 \ldots n, \ y_i \in \{0, 1\}\]

Classifier:

\[p_i = \sigma \left( w^T \phi(x_i) \right)\]

FIM for cross entropy loss for the last layer:

\[
\frac{\partial \ell}{\partial w} = \frac{1}{N} \sum_i (y_i - p_i) \phi(x_i)
\]

\[F_w = \frac{1}{N} \sum_i p_i (1 - p_i) \phi(x_i)\phi(x_i)^T\]

Two layer network

\[x \rightarrow \phi(x)\]
Properties of Task2vec embedding

Dataset:

\[ (x_i, y_i), i = 1 \ldots n, \quad y_i \in \{0, 1\} \]

Classifier:

\[ p_i = \sigma \left( w^T \phi(x_i) \right) \]

FIM for cross entropy loss for the last layer:

\[ \frac{\partial \ell}{\partial w} = \frac{1}{N} \sum_i (y_i - p_i) \phi(x_i) \]

\[ F_w = \frac{1}{N} \sum_i p_i (1 - p_i) \phi(x_i) \phi(x_i)^T \]

1. **Invariance** to label space
2. Encodes task **difficulty**
3. Encodes task **domain**
4. Encodes **useful features** for the task

**Representative domain embedding**

\[ D = \frac{1}{N} \sum_i \phi(x_i) \phi(x_i)^T \]
Properties of **Task2vec** embedding

1. Binary tasks on unit square, i.e., each tile is a task
2. 10 Random ReLU features, i.e., $\phi_i = \max(0, a_i x + b_i y + c_i)$
3. T-SNE to map 10x10 FIM to 2D
Properties of Task2vec embedding

1. Binary tasks on unit square, i.e., each tile is a task
2. 10 Random ReLU features, i.e., $\phi_i = \max(0, a_i x + b_i y + c_i)$
3. T-SNE to map 10x10 FIM to 2D

Polynomial degree 3
Robust Fisher Computation

1. For realistic CV tasks we want to use deep CNNs (e.g., ResNet) and estimate FIM for all the parameters.

2. **Challenge:** FIM can be hard to estimate (noisy loss landscape; high dimensions; small training set)

3. Robust FIM
   1. Restrict it to a diagonal
   2. Restrict it a single value per filter (CNN layer)
   3. Robust estimation via perturbation

Estimate $\Lambda$ of a Gaussian perturbation:

$$L(\hat{w}; \Lambda) = \mathbb{E}_{w \sim \mathcal{N}(\hat{w}, \Lambda)} [H_{p_w, \hat{p}} p(y|x)] + \beta KL(\mathcal{N}(0, \Lambda) \| \mathcal{N}(0, \lambda^2 I))$$

Optimal $\Lambda$ satisfies:

$$\frac{\beta}{2N} \Lambda = F + \frac{\beta \lambda^2}{2N} I$$

“Trivial Embedding”
Similarity measures on the space of tasks

Task = \{dataset, labels, loss\}
Similarity measures on the space of tasks

Domain similarity

Unbiased look at dataset bias, Torralba and Efros, CVPR 11
Similarity measures on the space of tasks

Domain similarity

Range / label similarity

• e.g., Taxonomic distance

$$D_{\text{tax}}(t_a, t_b) = \min_{i \in S_a, j \in S_b} d(i, j).$$

https://www.pinterest.com/pin/520799144386337065/
Similarity measures on the space of tasks

Domain similarity

Range / label similarity
  • e.g., Taxonomic distance

$$D_{\text{tax}}(t_a, t_b) = \min_{i \in S_a, j \in S_b} d(i, j).$$

Transfer “distance”
  • Fine-tune on a followed by b

$$D_{ft}(t_a \rightarrow t_b) = \frac{\mathbb{E}[\ell_{a \rightarrow b}] - \mathbb{E}[\ell_b]}{\mathbb{E}[\ell_b]}$$

Taskonomy: Disentangling Task Transfer Learning, Amir Zamir, Alexander Sax, William Shen, Leonidas Guibas, Jitendra Malik, Silvio Savarese, CVPR 18
Distance measures on Task2vec embedding

Symmetric distance

\[ d_{\text{sym}}(F_a, F_b) = d_{\text{cos}}\left( \frac{F_a}{F_a + F_b}, \frac{F_b}{F_a + F_b} \right) \]

Asymmetric “distance”

\[ d_{\text{asym}}(t_a \rightarrow t_b) = d_{\text{sym}}(t_a, t_b) - \alpha d_{\text{sym}}(t_a, t_0) \]
**MODEL2VEC: Joint embedding of tasks and models**

1. So far we have been associating models (feature extractors) with the tasks they are trained on.

2. How about
   1. legacy / black-box feature extractors? E.g., SIFT, HOG, Fisher vector
   2. models of different complexity trained on the same dataset

3. **MODEL2VEC**: Jointly embed feature extractors (encoded as one-hot-vectors) and tasks such that similarity reflects a meta-task objective.
   1. Needs training data
Task Zoo

- Tasks [1460]
  - iNaturalist [207]
  - CUB 200 [25]
  - iMaterialist [228]
  - DeepFashion [1000]
Task Zoo

- Tasks [1460]
  - iNaturalist [207]
  - CUB 200 [25]
  - iMaterialist [228]
  - DeepFashion [1000]
Task Zoo

- Tasks [1460]
  - iNaturalist [207]
  - CUB 200 [25]
  - iMaterialist [228]
  - DeepFashion [1000]
Task Zoo

• Tasks [1460]
  • iNaturalist [207]
  • CUB 200 [25]
  • iMaterialist [228]
  • DeepFashion [1000]

• Few tasks > 10K training samples but most have 100-1000 samples
Experiment: Task2vec recapitulates iNaturalist taxonomy

Task embedding cosine similarity

ResNet trained on ImageNet as probe network
Experiment: Task2Vec norm encodes task difficulty

ResNet trained on ImageNet as probe network
Experiment: Task2vec vs Domain2vec

Task Embeddings
- Actinopterygii (n)
- Amphibia (n)
- Arachnida (n)
- Aves (n)
- Fungi (n)
- Insecta (n)
- Mammalia (n)
- Mollusca (n)
- Plantae (n)
- Protozoa (n)
- Reptilia (n)
- Category (m)
- Color (m)
- Gender (m)
- Material (m)

Domain Embeddings
- Carnivora
- Rodentia
- Laurales
- Liliales
- Pinales
- Rosales
- Black
- Green
- Blue
- Brown
- Purple
- Neckline (m)
- Pants (m)
- Pattern (m)
- Shoes (m)

Categories:
- Formal dresses
- Wedding dresses
- Prom dresses
- Yoga pants
- Sweatpants
- Denim
- Ripped
- Jeans
- Rodentia
- Carnivora
- Corvidae
- Falconiformes
- Passeriformes
- Corvida
Task and Feature Zoo

- Tasks [1460]
  - iNaturalist [207]
  - CUB 200 [25]
  - iMaterialist [228]
  - DeepFashion [1000]

- Feature Zoo [156 experts]
  - ResNet-34 pertained on ImageNet
  - Followed by fine-tuned on tasks with enough examples
The Matrix

Tasks

Feature extractors
The Matrix

iNaturalist + CUB

Tasks

Experts

[CUB] Laridae (aves)
[CUB] Thraupidae (aves)
[CUB] Bombycillidae (aves)
[CUB] Passeridae (aves)
[CUB] Prinipaliformes (aves)
[CUB] Caprimulgiformes (aves)
[CUB] Piciformes (aves)
[CUB] Anseriformes (aves)
[CUB] Pelecaniformes (aves)
[CUB] Loricifera (aves)
[CUB] Corvidae (aves)
[CUB] Podicipediformes (aves)
[CUB] Tyrannidae (aves)
[CUB] Apodiformes (aves)
[CUB] Hirundinidae (aves)
[CUB] Cuculiformes (aves)
[CUB] Monotremata (aves)
[CUB] Caprimulgiformes (aves)
[CUB] Charadriiformes (aves)
[CUB] Cardinales (aves)
[CUB] Trochilidae (aves)
[CUB] Charadriiformes (aves)
[Nat] Plocoptera (aves)
[Nat] Charadriiformes (aves)
[Nat] Anseriformes (aves)
[Nat] Anseriformes (aves)
[Nat] Hapalidae (aves)
[Nat] Accipitriformes (aves)
[Nat] Piciformes (aves)
[Nat] Columbiformes (aves)
[CUB] Vireonidae (aves)
[CUB] Tyrannidae (aves)
[CUB] Parulidae (aves)
[CUB] Emberizidae (aves)
[Nat] Sapindales (magnoliopsida)
[Nat] Caryophyllales (magnoliopsida)
[Nat] Rosales (magnoliopsida)
[Nat] Asterales (magnoliopsida)
[Nat] Lamiales (magnoliopsida)
[Nat] Fabales (magnoliopsida)
[Nat] Asparagales (ll已psida)
[Nat] Ranunculales (magnoliopsida)
[Nat] Gentianales (magnoliopsida)
[Nat] Ericales (magnoliopsida)
[Nat] Penicillium (actinomycetae)
[Nat] Rodentia (mammalia)
[Nat] Carnivora (mammalia)
[Nat] Squamata (reptilia)
[Nat] Anura (amphibia)
[Nat] Lepidoptera (insecta)
[Nat] Odonata (insecta)
[Nat] Coleoptera (insecta)
ImageNet expert is usually good but on many tasks the best expert handily outperforms the ImageNet expert.
Data efficiency of Task2Vec

![Graph showing data efficiency of different methods.](image)

- ImageNet fixed
- Task2vec fixed
- Brute force fixed
- ImageNet finetune
- Task2vec finetune
### Choice of distance for **TASK2VEC**

<table>
<thead>
<tr>
<th>Meta-task</th>
<th>Optimal</th>
<th>Chance</th>
<th>ImageNet</th>
<th>TASK2VEC</th>
<th>Asymmetric TASK2VEC</th>
<th>MODEL2VEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>iNat + CUB</td>
<td>31.24</td>
<td>+59.52%</td>
<td>+30.18%</td>
<td>+42.54%</td>
<td>+9.97%</td>
<td>+6.81%</td>
</tr>
<tr>
<td>Mixed</td>
<td>22.90</td>
<td>+112.49%</td>
<td>+75.73%</td>
<td>+40.30%</td>
<td>+29.23%</td>
<td>+27.81%</td>
</tr>
</tbody>
</table>

Relative error increase over the oracle (best choice)
Choice of the probe network on **TASK2VEC**

<table>
<thead>
<tr>
<th>Probe network</th>
<th>Top-10</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chance</td>
<td>+13.95%</td>
<td>+59.52%</td>
</tr>
<tr>
<td>VGG-13</td>
<td>+4.82%</td>
<td>+38.03%</td>
</tr>
<tr>
<td>DenseNet-121</td>
<td>+0.30%</td>
<td>+10.63%</td>
</tr>
<tr>
<td>ResNet-13</td>
<td>+0.00%</td>
<td>+9.97%</td>
</tr>
</tbody>
</table>

Relative error increase over the oracle (best choice)
Thank you!