Artificial Intelligence

The Theory of Learning and Ensemble Learning

Min-Yuh Day, Ph.D, Associate Professor

Institute of Information Management, National Taipei University

https://web.ntpu.edu.tw/~myday
## Syllabus

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Subject/Topics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2022/09/14</td>
<td>Introduction to Artificial Intelligence</td>
</tr>
<tr>
<td>2</td>
<td>2022/09/21</td>
<td>Artificial Intelligence and Intelligent Agents</td>
</tr>
<tr>
<td>3</td>
<td>2022/09/28</td>
<td>Problem Solving</td>
</tr>
<tr>
<td>4</td>
<td>2022/10/05</td>
<td>Knowledge, Reasoning and Knowledge Representation; Uncertain Knowledge and Reasoning</td>
</tr>
<tr>
<td>5</td>
<td>2022/10/12</td>
<td>Case Study on Artificial Intelligence I</td>
</tr>
<tr>
<td>6</td>
<td>2022/10/19</td>
<td>Machine Learning: Supervised and Unsupervised Learning</td>
</tr>
<tr>
<td>Week</td>
<td>Date</td>
<td>Subject/Topics</td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
<td>-----------------------------------------------------</td>
</tr>
<tr>
<td>7</td>
<td>2022/10/26</td>
<td>The Theory of Learning and Ensemble Learning</td>
</tr>
<tr>
<td>8</td>
<td>2022/11/02</td>
<td>Midterm Project Report</td>
</tr>
<tr>
<td>9</td>
<td>2022/11/09</td>
<td>Deep Learning and Reinforcement Learning</td>
</tr>
<tr>
<td>10</td>
<td>2022/11/16</td>
<td>Deep Learning for Natural Language Processing</td>
</tr>
<tr>
<td>11</td>
<td>2022/11/23</td>
<td>Invited Talk: AI for Information Retrieval</td>
</tr>
<tr>
<td>12</td>
<td>2022/11/30</td>
<td>Case Study on Artificial Intelligence II</td>
</tr>
<tr>
<td>Week</td>
<td>Date</td>
<td>Subject/Topics</td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>13</td>
<td>2022/12/07</td>
<td>Computer Vision and Robotics</td>
</tr>
<tr>
<td>14</td>
<td>2022/12/14</td>
<td>Philosophy and Ethics of AI and the Future of AI</td>
</tr>
<tr>
<td>15</td>
<td>2022/12/21</td>
<td>Final Project Report I</td>
</tr>
<tr>
<td>16</td>
<td>2022/12/28</td>
<td>Final Project Report II</td>
</tr>
<tr>
<td>17</td>
<td>2023/01/04</td>
<td>Self-learning</td>
</tr>
<tr>
<td>18</td>
<td>2023/01/11</td>
<td>Self-learning</td>
</tr>
</tbody>
</table>
The Theory of Learning and Ensemble Learning
• The Theory of Learning
  • Computational Learning Theory
  • Probably Approximately Correct (PAC) Learning
• Ensemble Learning
  • Bagging: Random forests
  • Stacking
  • Boosting: Gradient boosting
  • Online learning
• Meta Learning: Learning to Learn
Artificial Intelligence: A Modern Approach

1. Artificial Intelligence
2. Problem Solving
3. Knowledge and Reasoning
4. Uncertain Knowledge and Reasoning
5. Machine Learning
6. Communicating, Perceiving, and Acting
7. Philosophy and Ethics of AI

Artificial Intelligence: Machine Learning
Artificial Intelligence: 5. Machine Learning

- Learning from Examples
- Learning Probabilistic Models
- Deep Learning
- Reinforcement Learning

Reinforcement Learning (DL)

Agent

Environment

Reinforcement Learning (DL)

1. Observation
2. Action
3. Reward

Reinforcement Learning (DL)

Agents interact with environments through sensors and actuators

Figure 2.1

Figure 2.2 A vacuum world with just two locations. Each location can be clean or dirty, and the agent can move left or right and can clean the square that it occupies. Different versions of the vacuum world allow for different rules about what the agent can perceive, whether its actions always succeed, and so on.

Machine Learning
Supervised Learning (Classification)
Learning from Examples

\[ y = f(x) \]
Machine Learning
Supervised Learning (Classification)
Learning from Examples

\[ y = f(x) \]

*input*  \hspace{1cm}  *Output*  
\hspace{1cm}  *label*
Machine Learning
Supervised Learning (Classification)
Learning from Examples

\[ y = f(x) \]

5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.9,3.1,4.9,1.5,Iris-versicolor
6.3,3.3,6.0,2.5,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
7.1,3.0,5.9,2.1,Iris-virginica
Machine Learning
Supervised Learning (Classification)
Learning from Examples

\[ y = f(x) \]

Example

5.1, 3.5, 1.4, 0.2, Iris-setosa
4.9, 3.0, 1.4, 0.2, Iris-setosa
4.7, 3.2, 1.3, 0.2, Iris-setosa
7.0, 3.2, 4.7, 1.4, Iris-versicolor
6.4, 3.2, 4.5, 1.5, Iris-versicolor
6.9, 3.1, 4.9, 1.5, Iris-versicolor
6.3, 3.3, 6.0, 2.5, Iris-virginica
5.8, 2.7, 5.1, 1.9, Iris-virginica
7.1, 3.0, 5.9, 2.1, Iris-virginica
Machine Learning
Supervised Learning (Classification)
Learning from Examples

\[ y = f(x) \]

Example

<table>
<thead>
<tr>
<th>( x )</th>
<th>( y )</th>
<th>Species</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1, 3.5, 1.4, 0.2</td>
<td>Iris-setosa</td>
<td></td>
</tr>
<tr>
<td>4.9, 3.0, 1.4, 0.2</td>
<td>Iris-setosa</td>
<td></td>
</tr>
<tr>
<td>4.7, 3.2, 1.3, 0.2</td>
<td>Iris-setosa</td>
<td></td>
</tr>
<tr>
<td>7.0, 3.2, 4.7, 1.4</td>
<td>Iris-versicolor</td>
<td></td>
</tr>
<tr>
<td>6.4, 3.2, 4.5, 1.5</td>
<td>Iris-versicolor</td>
<td></td>
</tr>
<tr>
<td>6.9, 3.1, 4.9, 1.5</td>
<td>Iris-versicolor</td>
<td></td>
</tr>
<tr>
<td>6.3, 3.3, 6.0, 2.5</td>
<td>Iris-virginica</td>
<td></td>
</tr>
<tr>
<td>5.8, 2.7, 5.1, 1.9</td>
<td>Iris-virginica</td>
<td></td>
</tr>
<tr>
<td>7.1, 3.0, 5.9, 2.1</td>
<td>Iris-virginica</td>
<td></td>
</tr>
</tbody>
</table>
Artificial Intelligence

Machine Learning & Deep Learning

Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Artificial Intelligence (AI)

Machine Learning (ML)

- Supervised Learning
- Unsupervised Learning
- Semi-supervised Learning
- Reinforcement Learning

Deep Learning (DL)

- CNN
- RNN
- LSTM
- GRU
- GAN

Source: https://leonardoaraujosantos.gitbooks.io/artificial-intelligence/content/deep_learning.html
The Theory of Learning
The Theory of Learning

• Computational Learning Theory
• Probably approximately correct (PAC)

The Theory of Learning

- How can we be sure that our learned hypothesis will predict well for previously unseen inputs?
  - How do we know that the hypothesis $h$ is close to the target function $f$ if we don’t know what is?
- How many examples do we need to get a good $h$?
- What hypothesis space should we use?
- If the hypothesis space is very complex, can we even find the best $h$ or do we have to settle for a local maximum?
- How complex should $h$ be?
- How do we avoid overfitting?
Computational Learning Theory

• Intersection of AI, statistics, and theoretical computer science.

• Any hypothesis that is seriously wrong will almost certainly be “found out” with high probability after a small number of examples.

Probably approximately correct (PAC)

• Any **hypothesis** that is consistent with a sufficiently large set of training examples is unlikely to be seriously wrong.

• **PAC learning algorithm**:
  
  • Any learning algorithm that returns hypotheses that are probably approximately correct.

Linear function

\[ y = f(x) \]

\[ y = w_1 x + w_0 \]

\[ h_w(x) = w_1 x + w_0 \]
Linear Regression Weight Space

\[ h_w(x) = w_1 x + w_0 \]

\[ w^* = \arg\min_w \text{Loss}(h_w) \]

\[ y = 0.232 \, x + 246 \]

Loss function for Weights \((w_1, w_0)\)
Ensemble Learning
• Select a collection, or ensemble, of hypotheses, $h_1, h_2, \ldots, h_n$, and combine their predictions by averaging, voting, or by another level of machine learning.

Ensemble Models

Heterogeneous Ensemble

Source: Ramesh Sharda, Dursun Delen, and Efraim Turban (2017), Business Intelligence, Analytics, and Data Science: A Managerial Perspective, 4th Edition, Pearson
Ensemble Learning

• Base model
  • individual hypotheses
  • $h_1, h_2, \ldots, h_n$

• Ensemble model
  • hypotheses combination

Why Ensemble Learning

• Reduce bias
• Reduce variance

Ensemble Learning

- Bagging
  - Random forests
- Stacking
- Boosting
  - Gradient boosting
- Online learning

Ensemble Learning: Bagging

• **Bagging**
  • Generate distinct training sets by sampling with replacement from the original training set.

• **Classification:**
  • Plurality Vote (Majority Vote)

• **Regression:**
  • Average

Ensemble Learning: Random forests

• **Random forest** model is a form of decision tree bagging in which we take extra steps to make the ensemble of trees more diverse, to reduce variance.

• The key idea is to randomly vary the **attribute** choices (rather than the training examples)

Ensemble Learning: Random forests

• Extremely randomized trees (ExtraTrees)
  • Use randomness in selecting the split point value
  • for each selected attribute, we randomly sample several candidate values from a uniform distribution over the attribute’s range

Ensemble Learning: Stacking

• Staking
  • Stacked generalization combines multiple base models from different model classes trained on the same data.

• Bagging
  • Combines multiple base models of the same model class trained on different data.

Ensemble Learning: Boosting

• Boosting
  • The most popular ensemble method
• Weighted training set

Ensemble Learning: Boosting
Ensemble Learning: Gradient boosting

- Gradient boosting
  - Gradient boosting is a form of boosting using gradient descent
- Gradient boosting machines (GBM)
- Gradient boosted regression trees (GBRT)
- Popular method for regression and classification of factored tabular data

Ensemble Learning: Online learning

• Online learning
  • Data are not i.i.d. (independent and identically distributed)
  • An agent receives an input $x_i$ from nature, predicts the corresponding $y_i$ and then is told the correct answer.

Meta Learning: Learning to Learn
Deep Learning
Transfer Learning
Few-Shot Learning
Meta Learning
Deep Learning, Transfer Learning, Few-Shot Learning, Meta Learning

• Deep Learning
  • Transfer Learning
    • Pre-training, Fine-Tuning (FT)
• Meta Learning: Learning to Learn
• Few-Shot Learning (FSL)
• One-Shot Learning (1SL)
• Zero-Shot Learning (0SL)(ZSL)
Machine Learning, Deep Learning, Meta Learning

Machine Learning, Deep Learning, Meta Learning

(a) Signal processing

(b) Machine learning

(c) Deep learning

(d) Deep meta-learning

Few-Shot Learning (FSL) and Meta Learning

Machine learning from few training examples

## Meta Learning, Transfer Learning, Ensemble Learning, Continual Learning, Multi-Task Learning

<table>
<thead>
<tr>
<th>Features</th>
<th>Method</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning from prior experience</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Relationship between source tasks</td>
<td>No limitation</td>
<td>Related</td>
<td>Same</td>
<td>Task streams</td>
<td>Related</td>
</tr>
<tr>
<td>Relationship between source tasks and target tasks</td>
<td>No limitation</td>
<td>Related</td>
<td>Same</td>
<td>Related</td>
<td>Related</td>
</tr>
<tr>
<td>Considering the requirements of the target task</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Meta-Learning and Few-shot Learning
Notations and Terms

### Optimization-based Meta-learning

<table>
<thead>
<tr>
<th>Notation</th>
<th>Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_{train}^i$</td>
<td>Training set for task $T_i$</td>
</tr>
<tr>
<td>$D_{test}^i$</td>
<td>Test set for task $T_i$</td>
</tr>
<tr>
<td>$D_{meta-train}$</td>
<td>Meta-training set</td>
</tr>
<tr>
<td>$D_{meta-test}$</td>
<td>Meta-testing set</td>
</tr>
</tbody>
</table>

### Metric-based Meta-learning

<table>
<thead>
<tr>
<th>Notation</th>
<th>Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_i$</td>
<td>Support Set for task $T_i$</td>
</tr>
<tr>
<td>$Q_i$</td>
<td>Query Set for task $T_i$</td>
</tr>
<tr>
<td>$D_{train}$</td>
<td>Training Set</td>
</tr>
<tr>
<td>$D_{test}$</td>
<td>Test Set</td>
</tr>
</tbody>
</table>

# Meta-Learning Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_i$</td>
<td>Task $i$</td>
</tr>
<tr>
<td>$\mathcal{L}$</td>
<td>Loss function</td>
</tr>
<tr>
<td>$(x_k, y_k)$</td>
<td>Input-Output pair</td>
</tr>
<tr>
<td>$f_\theta$</td>
<td>Model (function) with parameters $\theta$</td>
</tr>
<tr>
<td>$g_{\theta_1}$</td>
<td>Embedding function</td>
</tr>
<tr>
<td>$d_{\theta_2}$ or $d$</td>
<td>Distance function</td>
</tr>
<tr>
<td>$g_\phi$</td>
<td>Meta-Learning model with parameters $\phi$</td>
</tr>
<tr>
<td>$P_\theta(y</td>
<td>x)$</td>
</tr>
<tr>
<td>$k_\theta(x_1, x_2)$</td>
<td>Kernel function measuring similarity between two vectors $x_1$ and $x_2$</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Softmax function</td>
</tr>
<tr>
<td>$\alpha, \beta$</td>
<td>Learning rates</td>
</tr>
<tr>
<td>$w$</td>
<td>Weights</td>
</tr>
<tr>
<td>$v_c$</td>
<td>Prototype of class $c$</td>
</tr>
<tr>
<td>$C$</td>
<td>Set of classes present in S</td>
</tr>
<tr>
<td>$S^c$</td>
<td>Subset of S containing all elements $(x_k, y_k)$ such that $y_k = c$</td>
</tr>
<tr>
<td>$\oplus$</td>
<td>Concatenation operator</td>
</tr>
<tr>
<td>$B$</td>
<td>Number of batches $(X_b, Y_b)$ sampled in inner-loop for a randomly sampled task $T_i$</td>
</tr>
<tr>
<td>$I$</td>
<td>Number of tasks $T_i$ sampled in inner-loop</td>
</tr>
<tr>
<td>$J$</td>
<td>Number of outer-loop iterations</td>
</tr>
</tbody>
</table>

Meta-Learning Example Setup

Few-Shot Learning (FSL)
Solving the FSL problem by meta-learning

Few-Shot Learning (FSL)

Meta-learning

Each task mimics the few-shot scenario, and can be completely non-overlapping. Support sets are used to train; query sets are used to evaluate the model.
Meta-Task Learning (MTL)
Transfer Learning Strategy for Meta-Learning

Meta Learning

The task construction of cross-domain transfer and domain generalization

Transfer Learning, Fine-tuning, Few-shot learning

- **Do you have labeled data?**
  - No ➡ Zero-shot learning
  - Yes ➡ How many labels?
    - A lot ➡ Fine-tune model
    - A few ➡ Do you have unlabeled data?
      - No ➡ - Embedding lookup
        - Few-shot learning
      - Yes ➡ - Domain adaption
        - UDA/UST

Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
Transfer Learning

Pre-training

Fine-Tuning

Transfer Learning
BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT (Bidirectional Encoder Representations from Transformers)

Overall pre-training and fine-tuning procedures for BERT

Meta Learning

Meta Learning

<table>
<thead>
<tr>
<th>Year</th>
<th>Achievement</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1987</td>
<td>(1) A new framework of “learning how to learn” with self-referential learning was proposed. The neural networks in self-referential learning can regard their weights as inputs and update them continuously. (2) Based on the conventional neural network, two types of weights were used to connect the neurons. Each type of weight presents a different learning speed.</td>
<td>[34,35]</td>
</tr>
<tr>
<td>1990</td>
<td>A synaptic learning rule, which is biologically plausible, was proposed to automatically study the learning rules.</td>
<td>[36]</td>
</tr>
<tr>
<td>1993</td>
<td>A chain of meta-networks was introduced to improve the learning capacity of a recurrent neural network for a dynamic environment.</td>
<td>[37]</td>
</tr>
<tr>
<td>1995</td>
<td>A framework was proposed to optimize the learning rule within a parametric learning rule space.</td>
<td>[38]</td>
</tr>
<tr>
<td>1996</td>
<td>An improved self-referential model was proposed. Time ratios were used to measure the effects of learning processes on the later learning processes.</td>
<td>[39]</td>
</tr>
<tr>
<td>1998</td>
<td>The term “Learning to learn” was proposed to equally represent the concept of meta-learning.</td>
<td>[40]</td>
</tr>
<tr>
<td>2001</td>
<td>Gradient descent methods were firstly used in meta-learning instead of evolutionary methods, which were widely used in previous research.</td>
<td>[41,42]</td>
</tr>
<tr>
<td>2003</td>
<td>A biologically plausible meta-reinforcement learning algorithm was proposed to tune the parameters of the meta-learning model dynamically and adaptively.</td>
<td>[43]</td>
</tr>
<tr>
<td>2004</td>
<td>A new perspective of meta-learning was proposed: exploring the interaction between the learning mechanism and the specific contexts to which the mechanism applies.</td>
<td>[9]</td>
</tr>
<tr>
<td>2008</td>
<td>The zero-data learning problem was addressed.</td>
<td>[44]</td>
</tr>
<tr>
<td>2013</td>
<td>The relationship between transfer learning and meta-learning was described.</td>
<td>[48]</td>
</tr>
<tr>
<td>2016</td>
<td>A meta-learning algorithm named gradient descent by gradient descent was proposed.</td>
<td>[49]</td>
</tr>
<tr>
<td>2017</td>
<td>(1) MAML was proposed. (2) A doctoral thesis systematically introduced the concept of meta-learning and corresponding methods.</td>
<td>[50,51]</td>
</tr>
<tr>
<td>2018</td>
<td>Reptile, an improved version of MAML, was proposed.</td>
<td>[52]</td>
</tr>
<tr>
<td>2019</td>
<td>The Capsule network provides a new method to improve the learning capacity of meta-learning, especially in computer vision.</td>
<td>[53]</td>
</tr>
<tr>
<td>2020</td>
<td>Combining auto-encoder and capsule network to focus on the zero-shot learning problem.</td>
<td>[54]</td>
</tr>
</tbody>
</table>

# Meta-learning Approaches

<table>
<thead>
<tr>
<th>Key idea</th>
<th>Metric-based</th>
<th>Optimization-based</th>
<th>Model-based</th>
</tr>
</thead>
<tbody>
<tr>
<td>**How (P_\theta(y</td>
<td>x)) is modeled?**</td>
<td>Metric Learning [19]</td>
<td>Gradient Descent</td>
</tr>
<tr>
<td>[ \sum_{(x_k, y_k) \in S} k_\theta(x, x_k) y_k, ]</td>
<td>[ P_\theta'(y</td>
<td>x), ]</td>
<td>[ f_\theta(x, S). ]</td>
</tr>
<tr>
<td>[ \text{where } \theta' = g_\phi(\theta, S) ]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Advantages | Faster Inference. | Offers flexibility to optimize in dynamic environments. \(S\) can be discarded post-optimization. | Faster inference with memory models. Eliminates the need for defining a metric or optimizing at test. |
| Disadvantages | Less adaptive to optimization in dynamic environments. | Optimization at inference is undesirable for real-world deployment. | Less efficient to hold data in memory as \(S\) grows. |
| | Easy to deploy. | Prone to overfitting. | Hard to design. |
| | Computational complexity grows linearly with size of \(S\) at test. | | |

# Meta Learning: Learning to Learn

<table>
<thead>
<tr>
<th>Class</th>
<th>Methods</th>
<th>Reference</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric-Based</td>
<td>Siamese Neural Networks</td>
<td>[32–36]</td>
<td>We show four metric-based meta-learning algorithms, focusing on feature extractors, similarity metrics, and automatic algorithm selection</td>
</tr>
<tr>
<td></td>
<td>Matching Networks</td>
<td>[37–41]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Prototype Networks</td>
<td>[42–46]</td>
<td>However, the metric-based approaches are sensitive to the dataset and</td>
</tr>
<tr>
<td></td>
<td>Relation Networks</td>
<td>[47–53]</td>
<td>increase the computational expenditure when the number of tasks is large.</td>
</tr>
<tr>
<td></td>
<td>Memory-Augmented Neural Networks</td>
<td>[54–56]</td>
<td>We display three model-based approaches. MANN combines neural networks with external memory modules, but the model is complex. Meta-Net is computationally intensive and has high memory requirements. SNAIL is relatively simplified, but has to be optimized in terms of automatic parameter tuning and reducing computation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[57,58]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Meta Networks</td>
<td>[59–65]</td>
<td></td>
</tr>
<tr>
<td>Model-Based</td>
<td>Simple Neural Attentive Meta-Learner</td>
<td>[66–71]</td>
<td>We present three methods of optimization-based meta-learning. MAML is relatively simple to implement, but the capacity of the model is limited.</td>
</tr>
<tr>
<td></td>
<td>MAML</td>
<td>[72–80]</td>
<td>META-LSTM has a large capacity, but a complicated training process. Meta-SGD has improved capacity but still has difficulties in generalization ability.</td>
</tr>
<tr>
<td></td>
<td>META-SGD</td>
<td>[87–93]</td>
<td></td>
</tr>
</tbody>
</table>

Metric-based Meta-learning

M-Way K-Shot Task (4-way-1-shot classification task)

# Metric-based Meta-Learning Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>T.I</th>
<th>$g_{\theta_1}$</th>
<th>$d_{\theta_2}$</th>
<th>Prediction</th>
<th>Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Siamese Networks [20]</td>
<td>Yes</td>
<td>CNN</td>
<td>L1</td>
<td>$v = w \cdot d(g_{\theta_1}(x_1), g_{\theta_2}(x_2))$</td>
<td>$-(y \log(p) + (1 - y)(\log(1 - p))$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$p = \text{sigmoid}(\sum_j v_j)$</td>
<td></td>
</tr>
<tr>
<td>Matching Networks [13]</td>
<td>Yes</td>
<td>CNN + LSTM w/ attention</td>
<td>Cosine Similarity</td>
<td>$\hat{y} = \sum_{k=1}^{t} \sigma(d(f_\theta(\hat{x}), g_{\theta_1}(x_k))y_k)$</td>
<td>$- \log P$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$P(y = c</td>
<td>x) = \hat{y}_c$</td>
</tr>
<tr>
<td>Prototypical Networks [21]</td>
<td>Yes</td>
<td>CNN</td>
<td>Euclidean</td>
<td>$P(y = c</td>
<td>x) = \sigma(-d(g_{\theta_1}(x), v_c))$</td>
</tr>
<tr>
<td>Relation Networks [22]</td>
<td>Yes</td>
<td>CNN</td>
<td>Learned by CNN</td>
<td>$r_c = d_{\theta_2}(g_{\theta_1}(x) \oplus v_c)$</td>
<td>$\sum_{c \in C} (r_c - 1(y == c))^2$</td>
</tr>
<tr>
<td>TADAM [16]</td>
<td>No</td>
<td>ResNet-12</td>
<td>Cosine / Euclidean</td>
<td>$P_\lambda(y = c</td>
<td>x) = \sigma(-\lambda d(g_{\theta_1}(x, \Gamma), v_c))$</td>
</tr>
<tr>
<td>TapNet [23]</td>
<td>No</td>
<td>Resnet-12</td>
<td>Euclidean</td>
<td>$P(y = c</td>
<td>x) = \sigma(-d(M(g_{\theta_1}(x)), M(\Phi_c)))$</td>
</tr>
<tr>
<td>CTM [24]</td>
<td>No</td>
<td>Any</td>
<td>Any</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Convolutional Siamese Network

Meta Learning: Matching Networks

Few-shot Prototypes

$\nu_c$ are computed as the mean of embedded support examples for each class

Meta Learning: Relation Network

Meta Learning: Category Traversal Module (CTM)

## Optimization-based Meta-Learning Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Learner</th>
<th>Meta-Learner</th>
</tr>
</thead>
</table>
| **LSTM Meta-Learner [35]** | Repeat $\forall b \in [1..B]$  
$L_b \leftarrow \mathcal{L}(f(X_b; \theta_{b-1}), Y_b)$  
$\theta_b \leftarrow g((\nabla \theta_b; L_b; \phi_{j-1}))$ | Repeat $\forall j \in [1..J]$  
$L_j^{\text{test}} \leftarrow \mathcal{L}(f(X; \theta_B), Y)$  
$\phi_j \leftarrow \phi_{j-1} - \alpha \nabla_{\phi_{j-1}} L_j^{\text{test}}$ |
| **MAML [14]** | Repeat $\forall i \in [1..I]$  
$L_i^{\text{train}} \leftarrow \mathcal{L}(f(D_i^{\text{train}}; \theta_{j-1}))$  
$\theta_i^{*} \leftarrow \theta_{j-1} - \alpha \nabla_{\theta_{j-1}} L_i^{\text{train}}$  
$L_i^{\text{test}} \leftarrow \mathcal{L}(f(D_i^{\text{test}}; \theta_i^{*}))$ | Repeat $\forall j \in [1..J]$  
$\theta_j \leftarrow \theta_{j-1} - \beta \nabla_{\theta_{j-1}} \sum_{i=1}^{I} L_i^{\text{test}}$  
$\phi_j \leftarrow \phi_{j-1} - \beta \nabla_{\phi_{j-1}} \sum_{i=1}^{I} L_i^{\text{test}}$ |
| **MTL [37]** | $L_i^{\text{train}} \leftarrow \mathcal{L}(f(D_i^{\text{train}}; \theta^{j-1}, \phi^{j-1}, \Theta))$  
$\theta_i^{*} \leftarrow \theta_{j-1} - \alpha \nabla_{\theta_{j-1}} L_i^{\text{train}}$  
$L_i^{\text{test}} \leftarrow \mathcal{L}(f(D_i^{\text{test}}; \theta_i^{*}))$ | $\theta_j \leftarrow \theta_{j-1} - \beta \nabla_{\theta_{j-1}} \sum_{i=1}^{I} L_i^{\text{test}}$  
$\phi_j \leftarrow \phi_{j-1} - \beta \nabla_{\phi_{j-1}} \sum_{i=1}^{I} L_i^{\text{test}}$ |
| **LEO [38]** | $\phi_{j-1} = \{\phi_e, \phi_r, \phi_d, \alpha\}$  
$z_i \leftarrow g(D_i^{\text{train}}; \phi_e, \phi_r, \Theta)$  
$\theta_i \leftarrow g(z_i; \phi_d)$  
$L_i^{\text{train}} \leftarrow \mathcal{L}(f(D_i^{\text{train}}; \theta_i))$  
$z_i^{*} \leftarrow z_i - \alpha \nabla_{z_i} L_i^{\text{train}}$  
$\theta_i^{*} \leftarrow g(z_i^{*}; \phi_d)$  
$L_i^{\text{test}} \leftarrow \mathcal{L}(f(D_i^{\text{test}}; \theta_i^{*}))$ | $\phi_j \leftarrow \phi_{j-1} - \beta \nabla_{\phi_{j-1}} \sum_{i=1}^{I} L_i^{\text{test}}$ |

Computational Graph for the Forward Pass of the Meta-learner

Model-Agnostic Meta-Learning (MAML)
Hierarchically Structured Meta-Learning (HSML)

Latent Embedding Optimization (LEO)

Overall Architecture of Meta Networks

The Transformers Timeline

Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
Pre-trained Models (PTM)

Pre-trained Models (PTM)

ULMFiT: 3 Steps
Transfer Learning in NLP

1. Pretraining
2. Domain adaptation
3. Fine-tuning

Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O’Reilly Media.
A typical pipeline for training transformer models
with the Datasets, Tokenizers, and Transformers libraries

Load and process datasets
Tokenize input texts
Load models, train and infer
Load metrics evaluate models

Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
https://github.com/nlp-with-transformers/notebooks
Few-Shot Learning (FSL)

Typical Scenarios

• Acting as a test bed for learning like human
• Learning for rare cases
• Reducing data gathering effort and computational cost

Few-Shot Learning (FSL)

• Few-Shot Learning (FSL) is a sub-area in machine learning.

• Machine Learning Definition

• A computer program is said to learn from experience $E$ with respect to some classes of task $T$ and performance measure $P$ if its performance can improve with $E$ on $T$ measured by $P$.

• Example: Image classification task ($T$), a machine learning program can improve its classification accuracy ($P$) through $E$ obtained by training on a large number of labeled images (e.g., the ImageNet data set).

## Machine Learning

<table>
<thead>
<tr>
<th>task $T$</th>
<th>experience $E$</th>
<th>performance $P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>image classification [73]</td>
<td>large-scale labeled images for each class</td>
<td>classification accuracy</td>
</tr>
<tr>
<td>the ancient game of Go [120]</td>
<td>a database containing around 30 million recorded moves of human experts and self-play records</td>
<td>winning rate</td>
</tr>
</tbody>
</table>

Few-Shot Learning (FSL)

• Few-shot Learning (FSL) is a type of machine learning problems (specified by E, T, and P), where E contains only a limited number of examples with supervised information for the target T.
  • Existing FSL problems are mainly supervised learning problems.
  • Few-shot classification learns classifiers given only a few labeled examples of each class.
    • image classification
    • sentiment classification from short text
    • object recognition

Few-Shot Learning (FSL)

• Few-shot classification learns a classifier \( h \), which predicts label \( y_i \) for each input \( x_i \).

• Usually, one considers the \textit{\( N \)-way-\( K \)-shot} classification, in which \( D_{\text{train}} \) contains \( I = KN \) examples from \( N \) classes each with \( K \) examples.

Few-Shot Learning (FSL)

• Few-Shot Learning (FSL)
  • $K = 10 \sim 100$ examples

• One-Shot Learning (1SL)
  • $K = 1$ example

• Zero-Shot Learning (0SL)(ZSL)
  • $K = 0$
## Few-Shot Learning (FSL)

<table>
<thead>
<tr>
<th>task $T$</th>
<th>experience $E$</th>
<th>performance $P$</th>
</tr>
</thead>
<tbody>
<tr>
<td>character generation [76]</td>
<td>a few examples of new character [supervised information]</td>
<td>pass rate of visual Turing test</td>
</tr>
<tr>
<td></td>
<td>pre-learned knowledge of parts and relations [prior knowledge]</td>
<td></td>
</tr>
<tr>
<td></td>
<td>similar molecules’ assays [prior knowledge]</td>
<td></td>
</tr>
<tr>
<td>image classification [70]</td>
<td>a few labeled images for each class of the target $T$ [supervised information]</td>
<td>classification accuracy [performance]</td>
</tr>
<tr>
<td></td>
<td>raw images of other classes, or pre-trained models [prior knowledge]</td>
<td></td>
</tr>
</tbody>
</table>

---

Few-Shot Learning (FSL)
Comparison of learning with sufficient and few training samples

Few-Shot Learning (FSL)
Different perspectives on how FSL methods solve the few-shot problem

(a) Data.  (b) Model.  (c) Algorithm.

Few-Shot Learning (FSL)

A taxonomy of FSL methods

Few-Shot Learning (FSL)

Few-Shot Learning (FSL)

Few-Shot Learning (FSL)

Few-Shot Learning (FSL)
Solving the FSL problem by data augmentation

# Few-Shot Learning (FSL)  
Characteristics for FSL Methods Focusing on the Data Perspective

<table>
<thead>
<tr>
<th>category</th>
<th>input $(x, y)$</th>
<th>transformer $t$</th>
<th>output $(\hat{x}, \hat{y})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>transforming samples from $D_{train}$</td>
<td>original $(x_i, y_i)$</td>
<td>learned transformation function on $x_i$</td>
<td>$(t(x_i), y_i)$</td>
</tr>
<tr>
<td>transforming samples from a weakly labeled or unlabeled data set</td>
<td>weakly labeled or unlabeled $(\bar{x}, -)$</td>
<td>a predictor trained from $D_{train}$</td>
<td>$(\bar{x}, t(\bar{x}))$</td>
</tr>
<tr>
<td>transforming samples from similar data sets</td>
<td>samples ${(\hat{x}_j, \hat{y}_j)}$ from similar data sets</td>
<td>an aggregator to combine ${(\hat{x}_j, \hat{y}_j)}$</td>
<td>$(t({\hat{x}_j}), t({\hat{y}_j}))$</td>
</tr>
</tbody>
</table>

The transformer $t(\cdot)$ takes input $(x, y)$ and returns synthesized sample $(\hat{x}, \hat{y})$ to augment the few-shot $D_{train}$.

### Few-Shot Learning (FSL)
#### Characteristics for FSL Methods Focusing on the Model Perspective

<table>
<thead>
<tr>
<th>strategy</th>
<th>prior knowledge</th>
<th>how to constrain $\mathcal{H}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>multitask learning</td>
<td>other $T$’s with their data sets $D$’s</td>
<td>share/tie parameter</td>
</tr>
<tr>
<td>embedding learning</td>
<td>embedding learned from/together with other $T$’s</td>
<td>project samples to a smaller embedding space in which similar and dissimilar samples can be easily discriminated</td>
</tr>
<tr>
<td>learning with external memory</td>
<td>embedding learned from other $T$’s to interact with memory</td>
<td>refine samples using key-value pairs stored in memory</td>
</tr>
<tr>
<td>generative modeling</td>
<td>prior model learned from other $T$’s</td>
<td>restrict the form of distribution</td>
</tr>
</tbody>
</table>

Few-Shot Learning (FSL)
Solving the FSL problem by multitask learning with parameter sharing

Few-Shot Learning (FSL)
Solving the FSL problem by multitask learning with parameter tying
# Few-Shot Learning (FSL)

## Characteristics of Embedding Learning Methods

<table>
<thead>
<tr>
<th>category</th>
<th>method</th>
<th>embedding function $f$ for $x_{test}$</th>
<th>embedding function $g$ for $D_{train}$</th>
<th>similarity measure $s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>task-specific</td>
<td>mAP-DLM/SSVM[130]</td>
<td>CNN</td>
<td>the same as $f$</td>
<td>cosine similarity</td>
</tr>
<tr>
<td></td>
<td>class relevance pseudo-metric [36]</td>
<td>kernel</td>
<td>the same as $f$</td>
<td>squared $\ell_2$ distance</td>
</tr>
<tr>
<td></td>
<td>convolutional siamese net [70]</td>
<td>CNN</td>
<td>the same as $f$</td>
<td>weighted $\ell_1$ distance</td>
</tr>
<tr>
<td></td>
<td>Micro-Set[127]</td>
<td>logistic projection</td>
<td>the same as $f$</td>
<td>$\ell_2$ distance</td>
</tr>
<tr>
<td></td>
<td>Matching Nets [138]</td>
<td>CNN, LSTM</td>
<td>CNN, biLSTM</td>
<td>cosine similarity</td>
</tr>
<tr>
<td></td>
<td>resLSTM [4]</td>
<td>GNN, LSTM</td>
<td>GNN, LSTM</td>
<td>cosine similarity</td>
</tr>
<tr>
<td></td>
<td>Active MN [8]</td>
<td>CNN</td>
<td>biLSTM</td>
<td>cosine similarity</td>
</tr>
<tr>
<td></td>
<td>SSMN [24]</td>
<td>CNN</td>
<td>another CNN</td>
<td>learned distance</td>
</tr>
<tr>
<td></td>
<td>ProtoNet [121]</td>
<td>CNN</td>
<td>the same as $f$</td>
<td>squared $\ell_2$ distance</td>
</tr>
<tr>
<td></td>
<td>semi-supervised ProtoNet[108]</td>
<td>CNN</td>
<td>the same as $f$</td>
<td>squared $\ell_2$ distance</td>
</tr>
<tr>
<td></td>
<td>PMN [141]</td>
<td>CNN, LSTM</td>
<td>CNN, biLSTM</td>
<td>cosine similarity</td>
</tr>
<tr>
<td></td>
<td>ARC [119]</td>
<td>LSTM, biLSTM</td>
<td>the same as $f$</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Relation Net [126]</td>
<td>CNN</td>
<td>the same as $f$</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>GNN [115]</td>
<td>CNN, GNN</td>
<td>the same as $f$</td>
<td>learned distance</td>
</tr>
<tr>
<td></td>
<td>TPN [84]</td>
<td>CNN</td>
<td>the same as $f$</td>
<td>Gaussian similarity</td>
</tr>
<tr>
<td></td>
<td>SNAIL [91]</td>
<td>CNN</td>
<td>the same as $f$</td>
<td>-</td>
</tr>
<tr>
<td>hybrid</td>
<td>Learnet [14]</td>
<td>adaptive CNN</td>
<td>CNN</td>
<td>weighted $\ell_1$ distance</td>
</tr>
<tr>
<td></td>
<td>DCCN [162]</td>
<td>adaptive CNN</td>
<td>CNN</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>R2-D2 [13]</td>
<td>adaptive CNN</td>
<td>CNN</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>TADAM [100]</td>
<td>adaptive CNN</td>
<td>the same as $f$</td>
<td>squared $\ell_2$ distance</td>
</tr>
</tbody>
</table>

Few-Shot Learning (FSL)
Solving the FSL problem by task-invariant embedding model

Few-Shot Learning (FSL)
Solving the FSL problem by hybrid embedding model

# Few-Shot Learning (FSL)
Solving the FSL problem by learning with external memory

![Diagram showing few-shot learning process]

### Few-Shot Learning (FSL)

#### Characteristics of FSL Methods Based on Learning with External Memory

<table>
<thead>
<tr>
<th>Category</th>
<th>Method</th>
<th>Memory $M$</th>
<th>Similarity $s$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>key $M_{\text{key}}$</td>
<td>value $M_{\text{value}}$</td>
</tr>
<tr>
<td><strong>refining representations</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MANN [114]</td>
<td>$f(x_i, y_{i-1})$</td>
<td>$f(x_i, y_{i-1})$</td>
<td>cosine similarity</td>
</tr>
<tr>
<td>APL [104]</td>
<td>$f(x_i)$</td>
<td>$y_i$</td>
<td>squared $\ell_2$ distance</td>
</tr>
<tr>
<td>abstraction memory [149]</td>
<td>$f(x_i)$</td>
<td>word embedding of $y_i$</td>
<td>dot product</td>
</tr>
<tr>
<td>CMN [164]</td>
<td>$f(x_i)$</td>
<td>$y_i$, age</td>
<td>dot product</td>
</tr>
<tr>
<td>life-long memory [65]</td>
<td>$f(x_i)$</td>
<td>$y_i$, age</td>
<td>cosine similarity</td>
</tr>
<tr>
<td>Mem2Vec [125]</td>
<td>$f(x_i)$</td>
<td>word embedding of $y_i$, age</td>
<td>dot product</td>
</tr>
<tr>
<td><strong>refining parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MetaNet [96]</td>
<td>$f(x_i)$</td>
<td>fast weight</td>
<td>cosine similarity</td>
</tr>
<tr>
<td>CSNs [97]</td>
<td>$f(x_i)$</td>
<td>fast weight</td>
<td>cosine similarity</td>
</tr>
<tr>
<td>MN-Net [22]</td>
<td>$f(x_i)$</td>
<td>$y_i$</td>
<td>dot product</td>
</tr>
</tbody>
</table>

Here, $f$ is an embedding function usually pre-trained by CNN or LSTM.

Few-Shot Learning (FSL)
Solving the FSL problem by generative modeling
Few-Shot Learning (FSL)
Solving the FSL problem by fine-tuning existing parameter $\theta_0$ by regularization

# Few-Shot Learning (FSL)

**Characteristics for FSL Methods**

*Focusing on the Algorithm Perspective*

<table>
<thead>
<tr>
<th>strategy</th>
<th>prior knowledge</th>
<th>how to search $\theta$ of the $h^*$ in $\mathcal{H}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>refining existing parameters</td>
<td>learned $\theta_0$</td>
<td>refine $\theta_0$ by $D_{\text{train}}$</td>
</tr>
<tr>
<td>refining meta-learned parameters</td>
<td>meta-learner</td>
<td>refine $\theta_0$ by $D_{\text{train}}$</td>
</tr>
<tr>
<td>learning the optimizer</td>
<td>meta-learner</td>
<td>use search steps provided by the meta-learner</td>
</tr>
</tbody>
</table>

Few-Shot Learning (FSL)
Solving the FSL problem by meta-learning

Few-Shot Learning (FSL)

Meta-learning

Each task mimics the few-shot scenario, and can be completely non-overlapping. Support sets are used to train; query sets are used to evaluate the model.

Few-Shot Learning (FSL)
Matching networks

Support Set (S)

<table>
<thead>
<tr>
<th>label</th>
<th>instance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addicted</td>
<td>Lately I’ve been splurging on crack/cocaine.</td>
</tr>
<tr>
<td>Recovery</td>
<td>I’m an addict in recovery, 2.5 years clean from drugs.</td>
</tr>
<tr>
<td>Others</td>
<td>I write a blog, about addiction, depression, and anxiety.</td>
</tr>
</tbody>
</table>

Query Set

Yesterday I snorted some dope that most likely had fentanyl in it and I nearly died.

Few-Shot Learning (FSL)

Prototypical network

# Few-Shot Learning (FSL) for medical text

<table>
<thead>
<tr>
<th>Study</th>
<th>Year</th>
<th>Data source</th>
<th>Research aim</th>
<th>Size of training set</th>
<th>Number of entities / classes</th>
<th>Entity type of training domain</th>
<th>Entity type of test domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alicia Lan-Clares and Ana Garcia-Serrano</td>
<td>2019</td>
<td>MEDDOCAN shared task dataset</td>
<td>NER</td>
<td>500 clinical cases, with no reconstruction</td>
<td>29</td>
<td>Clinical</td>
<td>Clinical</td>
</tr>
<tr>
<td>Ferré et al.</td>
<td>2019</td>
<td>BB-norm dataset from the Bacteria Biotpe 2019 Task</td>
<td>Entity Normalization</td>
<td>Original dataset with no reconstruction and zero-shot</td>
<td>Not mentioned †</td>
<td>Biological</td>
<td>Biological</td>
</tr>
<tr>
<td>Hou et al.</td>
<td>2020</td>
<td>Snips dataset</td>
<td>Slot Tagging (NER)</td>
<td>1-shot and 5-shot</td>
<td>7</td>
<td>Six of Weather, Music, Playlist, Book (including biomedical), Search Screen (including biomedical), Restaurant and Creative Work.</td>
<td>The remaining one</td>
</tr>
<tr>
<td>Sharaf et al.</td>
<td>2020</td>
<td>ten different datasets collected from the Open Parallel Corpus (OPUS)</td>
<td>Neural Machine Translation (NMT)</td>
<td>Sizes ranging from 4k to 64k training words (200 to 3200 sentences), but reconstructed</td>
<td>N/A †</td>
<td>Bible, European Central Bank, KDE, Quran, WMT news test sets, Books, European Medicines Agency (EMEA), Global Voices, Medical (ufal-Med), TED talks</td>
<td>Bible, European Central Bank, KDE, Quran, WMT news test sets, Books, European Medicines Agency (EMEA), Global Voices, Medical (ufal-Med), TED talks</td>
</tr>
<tr>
<td>Lu et al.</td>
<td>2020</td>
<td>MIMIC II and MIMIC III, and EU legislation dataset</td>
<td>Multi-label Text Classification</td>
<td>5-shot for MIMIC II and III, 50-shot for EU legislation</td>
<td>MIMIC II: 9 MIMIC III: 15 EU legislation: 5</td>
<td>Medical</td>
<td>Medical</td>
</tr>
</tbody>
</table>

# Few-Shot Learning (FSL) for medical text

<table>
<thead>
<tr>
<th>Study</th>
<th>Year</th>
<th>Data source</th>
<th>Research aim</th>
<th>Size of training set</th>
<th>Number of entities/classes</th>
<th>Entity type of training domain</th>
<th>Entity type of test domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lu et al.</td>
<td>2021</td>
<td>Constructed and shared a novel dataset based on Weibo for the research of few-shot rumor detection, and use PHEME dataset.</td>
<td>Rumor Detection (NER)</td>
<td>For the Weibo dataset: 2-way 3-event 5-shot 9-query; for PHEME dataset: 2-way 2-event 5-shot 9-query</td>
<td>Weibo: 14; PHEME: 5</td>
<td>Source posts and comments from Sina Weibo related to COVID-19</td>
<td>Source posts and comments from Sina Weibo related to COVID-19</td>
</tr>
<tr>
<td>Ma et al.</td>
<td>2021</td>
<td>CCLE, CERES-correction, CRISPR gene disruption scores, GDS10000 dataset, IPDTC dataset and FDX dataset.</td>
<td>Drug-response Predictions</td>
<td>1-shot, 2-shot, 5-shot and 10-shot</td>
<td>N/A</td>
<td>Biomedical</td>
<td>Biomedical</td>
</tr>
<tr>
<td>Kormilitzin et al.</td>
<td>2021</td>
<td>MIMIC-III and UK-CRIS datasets.</td>
<td>NER</td>
<td>25%, 50%, 75% and 100% of the training set, with no reconstruction</td>
<td>7</td>
<td>Electronic health record</td>
<td>Electronic health record</td>
</tr>
<tr>
<td>Guo et al.</td>
<td>2021</td>
<td>Abstracts of biomedical literature (from relation extraction task of BioNLP Shared Task 2011 and 2019) and structured biological datasets.</td>
<td>NER</td>
<td>100%, 75%, 50%, 25%, 9% of training set, with no reconstruction</td>
<td>Not mentioned</td>
<td>Biomedical entities</td>
<td>Biomedical entities</td>
</tr>
<tr>
<td>Lee et al.</td>
<td>2021</td>
<td>COVID-19-Scientific, COVID-19-Social (fact-checked by journalists from a website called Politi-fact.com), FEVER (Fact Extraction and Verification, generated by altering sentences extracted from Wikipedia to promote research online-checking systems).</td>
<td>Fact-Checking (close to Text Classification)</td>
<td>2-shot, 10-shot and 50-shot</td>
<td>Not mentioned</td>
<td>Facts about COVID-19</td>
<td>Facts about COVID-19</td>
</tr>
</tbody>
</table>

Curated samples with about five seeds required to get past well-known language model failure modes of either repeating text for the prompt or emitting text that does not pertain to the image. These samples demonstrate the ability to generate open-ended outputs that adapt to both images and text, and to make use of facts that it has learned during language-only pre-training.

Multimodal Few-Shot Learning with Frozen Language Models

Gradients through a frozen language model’s self attention layers are used to train the vision encoder.

Multimodal Few-Shot Learning with Frozen Language Models

Inference-Time interface for Frozen. The figure demonstrates how we can support (a) visual question answering, (b) outside-knowledge question answering and (c) few-shot image classification via in-context learning.

(a) 0-shot VQA
(b) 1-shot outside-knowledge VQA
(c) Few-shot image classification

Multimodal Few-Shot Learning with Frozen Language Models

Examples of (a) the Open-Ended miniImageNet evaluation (b) the Fast VQA evaluation.

GPT-3: Language Models are Few-Shot Learners

Language Models are Few-Shot Learners

Tom B. Brown*        Benjamin Mann*       Nick Ryder*       Melanie Subbiah*
Jared Kaplan†        Prafulla Dhariwal    Arvind Neelakantan Pranav Shyam
Girish Sastry        Amanda Askell        Sandhini Agarwal  Ariel Herbert-Voss
Gretchen Krueger     Tom Henighan         Rewon Child       Aditya Ramesh
Daniel M. Ziegler    Jeffrey Wu           Clemens Winter
Christopher Hesse    Mark Chen            Eric Sigler       Mateusz Litwin  Scott Gray
Benjamin Chess       Jack Clark           Christopher Berner
Sam McCandlish       Alec Radford         Ilya Sutskever    Dario Amodei

This work was funded by OpenAI. All models were trained on V100 GPU’s on part of a high-bandwidth cluster provided by Microsoft.

GPT-3: Language Models are Few-Shot Learners

GPT-3: Language Models are Few-Shot Learners

Performance on SuperGLUE increases with model size. A value of $K = 32$ means that our model was shown 32 examples per task, for 256 examples total divided across the 8 tasks in SuperGLUE.

GPT-3: Language Models are Few-Shot Learners

GPT-3: Language Models are Few-Shot Learners

Performance on cloze and completion tasks.

<table>
<thead>
<tr>
<th>Setting</th>
<th>LAMBADA (acc)</th>
<th>LAMBADA (ppl)</th>
<th>StoryCloze (acc)</th>
<th>HellaSwag (acc)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOTA</td>
<td>68.0&lt;sup&gt;a&lt;/sup&gt;</td>
<td>8.63&lt;sup&gt;b&lt;/sup&gt;</td>
<td>91.8&lt;sup&gt;c&lt;/sup&gt;</td>
<td>85.6&lt;sup&gt;d&lt;/sup&gt;</td>
</tr>
<tr>
<td>GPT-3 Zero-Shot</td>
<td>76.2</td>
<td>3.00</td>
<td>83.2</td>
<td>78.9</td>
</tr>
<tr>
<td>GPT-3 One-Shot</td>
<td>72.5</td>
<td>3.35</td>
<td>84.7</td>
<td>78.1</td>
</tr>
<tr>
<td>GPT-3 Few-Shot</td>
<td>86.4</td>
<td>1.92</td>
<td>87.7</td>
<td>79.3</td>
</tr>
</tbody>
</table>

GPT-3 significantly improves SOTA on LAMBADA while achieving respectable performance on two difficult completion prediction datasets.

GPT-3: Language Models are Few-Shot Learners

Results on three Open-Domain QA tasks

<table>
<thead>
<tr>
<th>Setting</th>
<th>NaturalQS</th>
<th>WebQS</th>
<th>TriviaQA</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAG (Fine-tuned, Open-Domain) [LPP+19]</td>
<td>44.5</td>
<td>45.5</td>
<td>68.0</td>
</tr>
<tr>
<td>T5-11B+SSM (Fine-tuned, Closed-Book) [RRS20]</td>
<td>36.6</td>
<td>44.7</td>
<td>60.5</td>
</tr>
<tr>
<td>T5-11B (Fine-tuned, Closed-Book)</td>
<td>34.5</td>
<td>37.4</td>
<td>50.1</td>
</tr>
<tr>
<td>GPT-3 Zero-Shot</td>
<td>14.6</td>
<td>14.4</td>
<td>64.3</td>
</tr>
<tr>
<td>GPT-3 One-Shot</td>
<td>23.0</td>
<td>25.3</td>
<td>68.0</td>
</tr>
<tr>
<td>GPT-3 Few-Shot</td>
<td>29.9</td>
<td>41.5</td>
<td>71.2</td>
</tr>
</tbody>
</table>

GPT-3 is shown in the few-, one-, and zero-shot settings, as compared to prior SOTA results for closed book and open domain settings. TriviaQA few-shot result is evaluated on the wiki split test server.

GPT-3: Language Models are Few-Shot Learners

GPT-3 results on a selection of QA / RC tasks.

<table>
<thead>
<tr>
<th>Setting</th>
<th>ARC (Easy)</th>
<th>ARC (Challenge)</th>
<th>CoQA</th>
<th>DROP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fine-tuned SOTA</td>
<td>92.0\textsuperscript{a}</td>
<td>78.5\textsuperscript{b}</td>
<td>90.7\textsuperscript{c}</td>
<td>89.1\textsuperscript{d}</td>
</tr>
<tr>
<td>GPT-3 Zero-Shot</td>
<td>68.8</td>
<td>51.4</td>
<td>81.5</td>
<td>23.6</td>
</tr>
<tr>
<td>GPT-3 One-Shot</td>
<td>71.2</td>
<td>53.2</td>
<td>84.0</td>
<td>34.3</td>
</tr>
<tr>
<td>GPT-3 Few-Shot</td>
<td>70.1</td>
<td>51.5</td>
<td>85.0</td>
<td>36.5</td>
</tr>
</tbody>
</table>

CoQA and DROP are F1 while ARC reports accuracy. See the appendix for additional experiments. a\textsuperscript{[KKS+20]} b\textsuperscript{[KKS+20]} c\textsuperscript{[JZC+19]} d\textsuperscript{[JN20]}

GPT-3: Language Models are Few-Shot Learners

<table>
<thead>
<tr>
<th>Setting</th>
<th>En→Fr</th>
<th>Fr→En</th>
<th>En→De</th>
<th>De→En</th>
<th>En→Ro</th>
<th>Ro→En</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOTA (Supervised)</td>
<td>45.6&lt;sup&gt;a&lt;/sup&gt;</td>
<td>35.0&lt;sup&gt;b&lt;/sup&gt;</td>
<td>41.2&lt;sup&gt;c&lt;/sup&gt;</td>
<td>40.2&lt;sup&gt;d&lt;/sup&gt;</td>
<td>38.5&lt;sup&gt;e&lt;/sup&gt;</td>
<td>39.9&lt;sup&gt;e&lt;/sup&gt;</td>
</tr>
<tr>
<td>XLM [LC19]</td>
<td>33.4</td>
<td>33.3</td>
<td>26.4</td>
<td>34.3</td>
<td>33.3</td>
<td>31.8</td>
</tr>
<tr>
<td>MASS [STQ+19]</td>
<td>37.5</td>
<td>34.9</td>
<td>28.3</td>
<td>35.2</td>
<td>35.2</td>
<td>33.1</td>
</tr>
<tr>
<td>mBART [LGG+20]</td>
<td>-</td>
<td>-</td>
<td>29.8</td>
<td>34.0</td>
<td>35.0</td>
<td>30.5</td>
</tr>
<tr>
<td>GPT-3 Zero-Shot</td>
<td>25.2</td>
<td>21.2</td>
<td>24.6</td>
<td>27.2</td>
<td>14.1</td>
<td>19.9</td>
</tr>
<tr>
<td>GPT-3 One-Shot</td>
<td>28.3</td>
<td>33.7</td>
<td>26.2</td>
<td>30.4</td>
<td>20.6</td>
<td>38.6</td>
</tr>
<tr>
<td>GPT-3 Few-Shot</td>
<td>32.6</td>
<td>39.2</td>
<td>29.7</td>
<td>40.6</td>
<td>21.0</td>
<td>39.5</td>
</tr>
</tbody>
</table>

Few-shot GPT-3 outperforms previous unsupervised NMT work by 5 BLEU when translating into English reflecting its strength as an English LM.

# GPT-3: Language Models are Few-Shot Learners

Performance of GPT-3 on SuperGLUE compared to fine-tuned baselines and SOTA

<table>
<thead>
<tr>
<th></th>
<th>SuperGLUE Average</th>
<th>BoolQ Accuracy</th>
<th>CB Accuracy</th>
<th>CB F1</th>
<th>COPA Accuracy</th>
<th>RTE Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fine-tuned SOTA</td>
<td>89.0</td>
<td>91.0</td>
<td>96.9</td>
<td>93.9</td>
<td>94.8</td>
<td>92.5</td>
</tr>
<tr>
<td>Fine-tuned BERT-Large</td>
<td>69.0</td>
<td>77.4</td>
<td>83.6</td>
<td>75.7</td>
<td>70.6</td>
<td>71.7</td>
</tr>
<tr>
<td>GPT-3 Few-Shot</td>
<td>71.8</td>
<td>76.4</td>
<td>75.6</td>
<td>52.0</td>
<td>92.0</td>
<td>69.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>WiC Accuracy</th>
<th>WSC Accuracy</th>
<th>MultiRC Accuracy</th>
<th>MultiRC F1a</th>
<th>ReCoRD Accuracy</th>
<th>ReCoRD F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fine-tuned SOTA</td>
<td>76.1</td>
<td>93.8</td>
<td>62.3</td>
<td>88.2</td>
<td>92.5</td>
<td>93.3</td>
</tr>
<tr>
<td>Fine-tuned BERT-Large</td>
<td>69.6</td>
<td>64.6</td>
<td>24.1</td>
<td>70.0</td>
<td>71.3</td>
<td>72.0</td>
</tr>
<tr>
<td>GPT-3 Few-Shot</td>
<td>49.4</td>
<td>80.1</td>
<td>30.5</td>
<td>75.4</td>
<td>90.2</td>
<td>91.1</td>
</tr>
</tbody>
</table>

GPT-3 few-shot is given a total of 32 examples within the context of each task and performs no gradient updates.

GPT-3: Language Models are Few-Shot Learners

GPT-3 3B is almost 10x larger than RoBERTa-Large (355M params), both models took roughly 50 petaflop/s-days of compute during pre-training.

# GPT-3: Language Models are Few-Shot Learners

Human accuracy in identifying whether short (~200 word) news articles are model generated

<table>
<thead>
<tr>
<th>Model</th>
<th>Mean accuracy</th>
<th>95% Confidence Interval (low, hi)</th>
<th>t compared to control (p-value)</th>
<th>“I don’t know” assignments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control (deliberately bad model)</td>
<td>86%</td>
<td>83%–90%</td>
<td>-</td>
<td>3.6%</td>
</tr>
<tr>
<td>GPT-3 Small</td>
<td>76%</td>
<td>72%–80%</td>
<td>3.9 (2e-4)</td>
<td>4.9%</td>
</tr>
<tr>
<td>GPT-3 Medium</td>
<td>61%</td>
<td>58%–65%</td>
<td>10.3 (7e-21)</td>
<td>6.0%</td>
</tr>
<tr>
<td>GPT-3 Large</td>
<td>68%</td>
<td>64%–72%</td>
<td>7.3 (3e-11)</td>
<td>8.7%</td>
</tr>
<tr>
<td>GPT-3 XL</td>
<td>62%</td>
<td>59%–65%</td>
<td>10.7 (1e-19)</td>
<td>7.5%</td>
</tr>
<tr>
<td>GPT-3 2.7B</td>
<td>62%</td>
<td>58%–65%</td>
<td>10.4 (5e-19)</td>
<td>7.1%</td>
</tr>
<tr>
<td>GPT-3 6.7B</td>
<td>60%</td>
<td>56%–63%</td>
<td>11.2 (3e-21)</td>
<td>6.2%</td>
</tr>
<tr>
<td>GPT-3 13B</td>
<td>55%</td>
<td>52%–58%</td>
<td>15.3 (1e-32)</td>
<td>7.1%</td>
</tr>
<tr>
<td>GPT-3 175B</td>
<td>52%</td>
<td>49%–54%</td>
<td>16.9 (1e-34)</td>
<td>7.8%</td>
</tr>
</tbody>
</table>

This table compares mean accuracy between five different models, and shows the results of a two-sample T-Test for the difference in mean accuracy between each model and the control model (an unconditional GPT-3 Small model with increased output randomness).

## GPT-3: Language Models are Few-Shot Learners

The GPT-3 generated news article that humans had the greatest difficulty distinguishing from a human written article (accuracy: 12%)

| Title: United Methodists Agree to Historic Split |
| Subtitle: Those who oppose gay marriage will form their own denomination |
| Article: After two days of intense debate, the United Methodist Church has agreed to a historic split - one that is expected to end in the creation of a new denomination, one that will be "theologically and socially conservative," according to The Washington Post. The majority of delegates attending the church’s annual General Conference in May voted to strengthen a ban on the ordination of LGBTQ clergy and to write new rules that will "discipline" clergy who officiate at same-sex weddings. But those who opposed these measures have a new plan: They say they will form a separate denomination by 2020, calling their church the Christian Methodist denomination. The Post notes that the denomination, which claims 12.5 million members, was in the early 20th century the "largest Protestant denomination in the U.S.," but that it has been shrinking in recent decades. The new split will be the second in the church’s history. The first occurred in 1968, when roughly 10 percent of the denomination left to form the Evangelical United Brethren Church. The Post notes that the proposed split "comes at a critical time for the church, which has been losing members for years," which has been "pushed toward the brink of a schism over the role of LGBTQ people in the church." Gay marriage is not the only issue that has divided the church. In 2016, the denomination was split over ordination of transgender clergy, with the North Pacific regional conference voting to ban them from serving as clergy, and the South Pacific regional conference voting to allow them. |

Transformer (Attention is All You Need)  
(Vaswani et al., 2017)

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

BERT (Bidirectional Encoder Representations from Transformers)

Overall pre-training and fine-tuning procedures for BERT

Transformer Models

Transformer

Encoder -> Decoder

- DistilBERT
- BERT
- RoBERTa
- XLM-R
- XLM
- ALBERT
- ELECTRA
- DeBERTa

- T5
- BART
- M2M-100
- BigBird

- GPT
- GPT-2
- CTRL
- GPT-3
- GPT-Neo
- GPT-J

Source: Lewis Tunstall, Leandro von Werra, and Thomas Wolf (2022), Natural Language Processing with Transformers: Building Language Applications with Hugging Face, O'Reilly Media.
Machine Learning: Ensemble Learning

Random Forest

https://tinyurl.com/aintpupython101
Machine Learning: Supervised Learning
Classification and Prediction

Machine Learning with scikit-learn

Classification and Prediction

```python
# Import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix

# Import sklearn
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.neural_network import MLPClassifier

print("Imported")

# Load dataset
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
```
Aurélien Géron (2019),
O’Reilly Media, 2019

https://github.com/ageron/handson-ml2

Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow

Notebooks
1. The Machine Learning landscape
2. End-to-end Machine Learning project
3. Classification
4. Training Models
5. Support Vector Machines
6. Decision Trees
7. Ensemble Learning and Random Forests
8. Dimensionality Reduction
9. Unsupervised Learning Techniques
10. Artificial Neural Nets with Keras
11. Training Deep Neural Networks
12. Custom Models and Training with TensorFlow
13. Loading and Preprocessing Data
14. Deep Computer Vision Using Convolutional Neural Networks
15. Processing Sequences Using RNNs and CNNs
16. Natural Language Processing with RNNs and Attention
17. Representation Learning Using Autoencoders
18. Reinforcement Learning
19. Training and Deploying TensorFlow Models at Scale

https://github.com/ageron/handson-ml2
Artificial Intelligence: A Modern Approach (AIMA)

• Artificial Intelligence: A Modern Approach (AIMA)
  • http://aima.cs.berkeley.edu/

• AIMA Python
  • http://aima.cs.berkeley.edu/python/readme.html
  • https://github.com/aimacode/aima-python

• Learning
  • http://aima.cs.berkeley.edu/python/learning.html


by Stuart Russell and Peter Norvig

The authoritative, most-used AI textbook, adopted by over 1500 schools.

Table of Contents for the US Edition (or see the Global Edition)

V Machine Learning
19 Learning from Examples ... 651
20 Learning Probabilistic Models ... 721
21 Deep Learning ... 750
22 Reinforcement Learning ... 789

VI Communicating, perceiving, and acting
23 Natural Language Processing ... 823
24 Deep Learning for Natural Language Processing ... 856
25 Computer Vision ... 881
26 Robotics ... 925

VII Conclusions
27 Philosophy, Ethics, and Safety of AI ... 981
28 The Future of AI ... 1012

Appendix A: Mathematical Background ... 1023
Appendix B: Notes on Languages and Algorithms ... 1030
Bibliography ... 1033 (pdf and LaTeX bibtex and bib data)
Index ... 1069 (pdf)

Exercises (website)
Figures (pdf)
Code (website); Pseudocode (pdf)
Covers: US, Global

http://aima.cs.berkeley.edu/
Papers with Code
State-of-the-Art (SOTA)

Browse State-of-the-Art

- 1509 leaderboards • 1327 tasks • 1347 datasets • 17810 papers with code

Follow on Twitter for updates

Computer Vision

- Semantic Segmentation: 33 leaderboards, 667 papers with code
- Image Classification: 62 leaderboards, 564 papers with code
- Object Detection: 54 leaderboards, 467 papers with code
- Image Generation: 81 leaderboards, 231 papers with code
- Pose Estimation: 40 leaderboards, 231 papers with code

› See all 707 tasks

Natural Language Processing

- Machine Translation
- Language Modelling
- Question Answering
- Sentiment Analysis
- Text Generation

https://paperswithcode.com/sota
Summary

• The Theory of Learning
  • Computational Learning Theory
  • Probably Approximately Correct (PAC) Learning
• Ensemble Learning
  • Bagging: Random forests
  • Stacking
  • Boosting: Gradient boosting
  • Online learning
• Meta Learning: Learning to Learn
References

• Steven D'Ascoli (2022), Artificial Intelligence and Deep Learning with Python: Every Line of Code Explained For Readers New to AI and New to Python, Independently published.
• Min-Yuh Day (2022), Python 101, https://tinyurl.com/aintpupython101