1. Introduction to Machine Learning Strategy
2. Setting Up your Goal
3. Comparing to Human-level Performance
4. Error Analysis
5. Mismatched Traing and Dev/Test Set
   1. Training and Testing on Different Distributions
   2. Bias and Variance with Mismatched Data Distributions
      * Ta thấy 1% Training Error và 10% Dev Error, thì nguyên nhân có thể như sau:
        + Variance
        + Data mismatch: thuật toán hoạt động bình thường chỉ là Dev set có distribution khác với Train set
      * Để khắc phục thì ta tạo **Training-dev set** có cùng distributiontừ Train set rồi xét bảng sau:

|  |  |  |
| --- | --- | --- |
| Training error | 1% | 1% |
| Training-dev error | 9% | 1.5% |
| Dev error | 10% | 10% |
|  | Variance | Data mismatch |

* + - Bằng cách so sánh Human level – Training set error – Training-dev set error – Dev error ta có thể quyết định vấn đề ở đâu:
      * Human level – Training set error: Avoidable Bias
      * Training set error – Training-dev set error: Variance
      * Training-dev error – Dev error: Data mismatch
    - Trong trường hợp Dev set dễ hơn Train set, ta có thể thấy Error giảm thay vì tăng
  1. Addressing Data Mismatch
     + Khi bị Data mismatch thì bạn nên kiếm thêm data cùng distribution với Dev set
     + 1 trong những cách làm: [**Artifact Data Synthesis**](https://www.google.com/search?sxsrf=ALiCzsaaIdWwfziY8z8u5wnEOlHhE2LpPg:1660042824261&q=Artifact+Data+Synthesis&spell=1&sa=X&ved=2ahUKEwiKkui2zbn5AhX3T2wGHSgxCSoQkeECKAB6BAgCEDo)

1. Learning from Multiple Tasks
   1. Transfer Learning
      * Ý tưởng: sử dụng model làm việc trên Task A, cho Task B (từ nhận diện động việc sang chuẩn đoán bệnh)
      * Cách làm: lấy model của Task A, rồi thay layer cuối của model ý thành một node output mới hoặc thêm layer. Sau đó train lại parameter của những layer mới đó bằng Data của Task B
      * Dùng khi:
        + Task A và B có cùng input
        + Task A có nhiều Data hơn Task B
        + Low level feature của A có thể ích cho B
   2. Multi-task Learning
      * Ý tưởng: train 1 model làm nhiều Task cùng lúc (nhận diện có xe, có biển báo, có vạch kẻ đường trong cùng 1 bức ảnh)
      * Cách làm: thay vì mỗi ảnh gán 1 nhãn, thì 1 ảnh gán nhiều nhãn (output *y* thành chiều)
      * Dùng khi nào:
        + Khi mà các task cùng chia sẻ low level feature
        + Số lượng data cho mỗi Task là tương đương nhau (gộp vào để có nhiều data để train hơn)
        + Có thể train 1 neural network đủ lớn
2. End-to-end Deep Learning
   1. End-to-end DL là gì?
      * Là sử dụng DL để từ đầu vào, trả vào đầu ra luôn thay vì phải thông qua nhiều Task khác nhau để đến được kết quả mong muốn.
      * Hình ảnh minh hoạ:  
        Graphical user interface, application, Word

        Description automatically generated
        + Trên: làm nhiều Task khác nhau để đến được kết quả mong muốn
        + Dưới: End-to-end
   2. Whether to use End-to-end Deep learning
      * Lợi:
        + Data tự nói lên chính nó
        + Không cần phải hand-design từng thành phần nhiều
      * Hại:
        + Cần nhiều dữ liệu
        + Có thể bỏ qua một số thành phần hand-design hữu dụng
      * Để áp dụng thì End-to-end DL thì câu hỏi mấu chốt là: **Bạn có đủ Data để cho model học một cái hàm đủ phức tạp để kết nối từ đầu vào X tới đầu ra Y**