Week 3: Data Labeling

CS 203: Software Tools and Techniques for AI

Prof. Nipun Batra
IIT Gandhinagar

The Story So Far

Week 1: Collected movie data from OMDb API

Week 2: Validated and cleaned the dataset

Now we have:

- Clean dataset with 50+ movies
- Features: title, year, genre, rating, director, etc.
- Data types validated, missing values handled

This week: Create labeled training data for ML tasks

The reality: Most ML models need labeled data to learn from

What is Data Labeling?

Data labeling: The process of annotating raw data with meaningful labels

Examples:

- Image classification: "This image contains a cat"
- Object detection: Draw boxes around cars in an image
- Sentiment analysis: "This review is positive"
- Named entity recognition: Mark person names in text

Why it matters: Supervised ML models learn from labeled examples

The Netflix Labeling Challenge

New scenario: Netflix wants to build ML models for:

1. **Movie poster classification**: Classify genre from poster images
2. **Review sentiment analysis**: Positive/negative user reviews
3. **Scene detection**: Identify action scenes in trailers
4. **Actor recognition**: Detect which actors appear in scenes

Each task needs labeled training data!

Today's Agenda

1. **Understanding Annotation Tasks** (types and challenges)
2. **Vision Tasks** (classification, detection, segmentation)
3. **Text Tasks** (classification, NER, sentiment)
4. **Label Studio** (annotation tool)
5. **Quality Metrics** (inter-annotator agreement)
6. **Best Practices** (building quality datasets)

Part 1: Understanding Annotation Tasks

What makes a good annotation task?

Types of Machine Learning Tasks

Classification: Assign category labels

- Image: "cat", "dog", "bird"
- Text: "positive", "negative", "neutral"

Detection: Find and locate objects

- Bounding boxes around objects
- Keypoints (facial landmarks)

Segmentation: Pixel-level labels

- Semantic: Label every pixel
- Instance: Separate object instances

Types of ML Tasks (continued)

Sequence Labeling: Tag each element

- Named Entity Recognition (NER)
- Part-of-speech tagging

Structured Prediction: Complex outputs

- Image captioning
- Question answering

Ranking: Order by relevance

- Search results
- Recommendations

Annotation Challenges

1. **Subjectivity**: Different annotators may disagree

- "Is this movie poster scary or thrilling?"
- Personal interpretation varies

2. **Ambiguity**: Data can be unclear

- Blurry images
- Ambiguous text

3. **Complexity**: Some tasks are inherently difficult

- Fine-grained classification (140 dog breeds)
- Multiple objects in one image

Annotation Challenges (continued)

4. **Cost**: Labeling is expensive

- Time-consuming manual work
- Expert knowledge sometimes required

5. **Scale**: ML needs large datasets

- ImageNet: 1.2 million labeled images
- Small datasets lead to overfitting

6. **Quality**: Errors propagate to the model

- Noisy labels hurt performance
- Need quality control measures

Part 2: Vision Annotation Tasks

Labeling images and videos

Image Classification

Task: Assign one label to an entire image

Example - Movie Poster Genre:

```
Image: [poster of Inception]
Label: Sci-Fi
```

Use cases:

- Content moderation
- Medical diagnosis
- Product categorization

Annotation: Simple - just pick a category

Multi-Label Classification

Task: Assign multiple labels to one image

Example - Movie Poster Genres:

```
Image: [poster of Inception]
Labels: ["Sci-Fi", "Thriller", "Action"]
```

Difference from single-label:

- One image can have multiple categories
- More flexible but more complex

Object Detection

Task: Find and locate objects with bounding boxes

Example - Actor Detection:

```
Image: [movie scene]
Boxes:
- [x:100, y:50, w:80, h:120] "Leonardo DiCaprio"
- [x:250, y:60, w:75, h:115] "Tom Hardy"
```

Output: Box coordinates + class label for each object

Use cases: Self-driving cars, face detection, security

Bounding Box Format

Common formats:

1. [x, y, width, height]: Top-left corner + dimensions

```
[100, 50, 80, 120]
```

2. [x1, y1, x2, y2]: Top-left + bottom-right corners

```
[100, 50, 180, 170]
```

3. [x_center, y_center, width, height]: YOLO format

```
[140, 110, 80, 120]
```

Semantic Segmentation

Task: Label every pixel with a class

Example - Scene Segmentation:

```
Pixel [0,0]: sky
Pixel [0,1]: sky
Pixel [100,200]: building
Pixel [150,300]: road
```

Use cases: Medical imaging, autonomous driving, satellite imagery

Annotation: Time-consuming - must trace object boundaries

Instance Segmentation

Task: Separate different instances of same class

Example - Multiple Actors:

```
Mask 1 (Person): Actor A's pixels
Mask 2 (Person): Actor B's pixels
```

Difference from semantic:

- Semantic: All actors labeled as "person"
- Instance: Each actor gets separate mask

More precise but more expensive to annotate

Keypoint Detection

Task: Mark specific points of interest

Example - Facial Landmarks:

```
Points:
- Left eye: (150, 100)
- Right eye: (200, 100)
- Nose: (175, 130)
- Mouth corners: (160, 160), (190, 160)
```

Use cases: Pose estimation, facial recognition, gesture control

Part 3: Text Annotation Tasks

Labeling text data

Text Classification

Task: Assign category to entire text

Example - Review Sentiment:

```
Text: "This movie was absolutely amazing!"
Label: Positive
```

Use cases:

- Spam detection
- Topic classification
- Sentiment analysis

Annotation: Read text, pick category

Named Entity Recognition (NER)

Task: Identify and classify entities in text

Example - Movie Review:

```
Text: "Christopher Nolan directed Inception in 2010."

Entities:
- "Christopher Nolan" [PERSON]
- "Inception" [MOVIE]
- "2010" [DATE]
```

Common entity types: PERSON, ORG, LOC, DATE, MONEY

NER Annotation Format

BIO Tagging (Beginning, Inside, Outside):

```
Christopher  B-PERSON
Nolan        I-PERSON
directed     O
Inception    B-MOVIE
in           O
2010         B-DATE
.            O
```

B-: Beginning of entity

I-: Inside entity (continuation)

O: Outside any entity

Span-Based NER

Alternative format: Character offsets

```
Text: "Christopher Nolan directed Inception"

Spans:
- [0, 17]: PERSON
- [27, 36]: MOVIE
```

Advantages:

- Handles nested entities
- Easier for some tools

Relation Extraction

Task: Identify relationships between entities

Example:

```
Text: "Christopher Nolan directed Inception."

Relations:
- (Christopher Nolan, directed, Inception)
- Type: DIRECTOR_OF
```

Use cases: Knowledge graphs, question answering

Text Sequence Labeling

Task: Label each token

Example - Part-of-Speech Tagging:

```
The      DET
movie    NOUN
was      VERB
great    ADJ
.        PUNCT
```

Other examples:

- Chunking (noun phrases, verb phrases)
- Slot filling (book flight to NYC on Monday)

Part 4: Label Studio

Open-source annotation tool

What is Label Studio?

Label Studio: Web-based annotation platform

Features:

- Multi-modal: Images, text, audio, video
- Customizable: Configure for any task
- Collaborative: Multiple annotators
- Export: ML-ready formats

Why use it?:

- Free and open-source
- Easy to set up
- Professional quality

Installing Label Studio

Option 1: pip install (recommended):

```
pip install label-studio
label-studio start
```

Option 2: Docker:

```
docker run -p 8080:8080 heartexlabs/label-studio
```

Access: Open browser to http://localhost:8080

Label Studio Interface

Main components:

1. **Projects**: Collection of annotation tasks
2. **Data**: Import your images/text
3. **Labeling Config**: Define annotation schema
4. **Annotators**: Assign work to people
5. **Export**: Download labeled data

Workflow: Create project → Import data → Configure labels → Annotate → Export

Creating a Project

Step 1: Click "Create Project"

Step 2: Give it a name ("Movie Poster Classification")

Step 3: Import data (upload images or provide URLs)

Step 4: Set up labeling interface

Step 5: Start annotating

Labeling Config: Image Classification

XML-based configuration:

```xml
<View>
<Image name="image" value="$image"/>
<Choices name="genre" toName="image">
<Choice value="Action"/>
<Choice value="Comedy"/>
<Choice value="Drama"/>
<Choice value="Horror"/>
<Choice value="Sci-Fi"/>
<Choice value="Romance"/>
</Choices>
</View>
```

This creates: Image viewer + genre selector

Labeling Config: Object Detection

For bounding boxes:

```
<View>
<Image name="image" value="$image"/>
<RectangleLabels name="label" toName="image">
<Label value="Actor"/>
<Label value="Text"/>
<Label value="Logo"/>
</RectangleLabels>
</View>
```

This creates: Image viewer + draw rectangle tool

Labeling Config: Text Classification

For sentiment analysis:

```
<View>
<Text name="text" value="$text"/>
<Choices name="sentiment" toName="text">
<Choice value="Positive"/>
<Choice value="Negative"/>
<Choice value="Neutral"/>
</Choices>
</View>
```

This creates: Text display + sentiment selector

Labeling Config: NER

For named entity recognition:

```
<View>
<Text name="text" value="$text"/>
<Labels name="label" toName="text">
<Label value="PERSON"/>
<Label value="MOVIE"/>
<Label value="DATE"/>
<Label value="ORGANIZATION"/>
</Labels>
</View>
```

This creates: Text viewer + highlight tool for entities

Importing Data

Format 1: JSON:

```json
[
  {
    "image": "https://example.com/poster1.jpg",
    "title": "Inception"
  },
  {
    "image": "https://example.com/poster2.jpg",
    "title": "The Matrix"
  }
]
```

Format 2: CSV:

```
image,title
poster1.jpg,Inception
poster2.jpg,The Matrix
```

Annotation Workflow

1. **Review task**: See the image/text
2. **Apply labels**: Click/select/draw
3. **Submit**: Save annotation
4. **Next task**: Move to next item

Keyboard shortcuts speed up annotation:

- Number keys: Select label
- Enter: Submit
- Skip: Skip difficult examples

Managing Annotators

Roles:

- **Annotator**: Labels data
- **Reviewer**: Checks quality
- **Manager**: Assigns tasks

Assignment strategies:

- Round-robin: Equal distribution
- Overlap: Multiple people label same item
- Expert: Hard cases go to best annotators

Exporting Labeled Data

Format options:

1. **JSON**: Full detail

```
{
"task": 1,
"result": [{
"value": {"choices": ["Sci-Fi"]},
"from_name": "genre"
}]
}
```

2. **CSV**: Simplified

```
image,genre
poster1.jpg,Sci-Fi
poster2.jpg,Action
```

Export Formats (continued)

3. COCO (for object detection):

```
{
"images": [...],
"annotations": [{
"image_id": 1,
"category_id": 1,
"bbox": [100, 50, 80, 120]
}],
"categories": [...]
}
```

4. YOLO (for detection):

```
0 0.5 0.5 0.2 0.3
1 0.7 0.6 0.15 0.25
```

Part 5: Quality Metrics

Measuring annotation quality

Why Measure Agreement?

Problem: Different annotators may disagree

Example:

```
Movie review: "The movie was okay."

Annotator A: Positive
Annotator B: Neutral
Annotator C: Negative
```

Need to measure: How much do annotators agree?

Goal: High agreement = clear task + good annotators

Simple Agreement

Percent agreement: How often annotators agree

Formula:

```
Agreement = (# agreements) / (# total items)
```

Example:

```
100 items, both annotators agree on 85

Agreement = 85 / 100 = 0.85 (85%)
```

Problem: Doesn't account for chance agreement

Cohen's Kappa

Better metric: Accounts for random chance

Formula:

```
κ = (p_o - p_e) / (1 - p_e)

p_o = observed agreement
p_e = expected agreement by chance
```

Range: -1 to 1

- $\kappa = 1$: Perfect agreement
- $\kappa = 0$: No agreement beyond chance
- $\kappa < 0$: Less than chance (bad!)

Cohen's Kappa Example

Data: 100 movie posters, 2 annotators

Results:

```
               Ann B: Action   Ann B: Drama
Ann A: Action        30             10
Ann A: Drama          5             55
```

Observed agreement: (30 + 55) / 100 = 0.85

Expected by chance:

- P(both say Action) = 0.40 × 0.35 = 0.14
- P(both say Drama) = 0.60 × 0.65 = 0.39
- p_e = 0.14 + 0.39 = 0.53

Kappa: (0.85 - 0.53) / (1 - 0.53) = 0.68

Interpreting Kappa

Guidelines (Landis & Koch):

- **< 0**: Poor
- **0.00 - 0.20**: Slight
- **0.21 - 0.40**: Fair
- **0.41 - 0.60**: Moderate
- **0.61 - 0.80**: Substantial
- **0.81 - 1.00**: Almost perfect

Our example: $\kappa = 0.68$ (Substantial agreement)

Calculating Kappa in Python

```python
from sklearn.metrics import cohen_kappa_score

# Annotations from two annotators
annotator_a = ['Action', 'Drama', 'Action', 'Drama', ...]
annotator_b = ['Action', 'Drama', 'Drama', 'Drama', ...]

kappa = cohen_kappa_score(annotator_a, annotator_b)
print(f"Cohen's Kappa: {kappa:.3f}")
```

Output: `Cohen's Kappa: 0.680`

Fleiss' Kappa

For 3+ annotators: Extension of Cohen's Kappa

Use case: Multiple annotators label same data

Formula: More complex, measures agreement across all annotators

Python:

```
from statsmodels.stats.inter_rater import fleiss_kappa

# Matrix: rows = items, cols = categories
# Values = count of annotators who chose category
data = [
[2, 1, 0],  # Item 1: 2 said Action, 1 said Drama
[0, 3, 0],  # Item 2: 3 said Drama
...
]

kappa = fleiss_kappa(data)
```

Confusion Matrix

Visualize disagreements:

```
                     Annotator B
                Action  Drama  Comedy
             Annotator A:
Action            30      8       2
Drama              5     45      10
Comedy             2      7      21
```

Diagonal: Agreements

Off-diagonal: Disagreements

Insights: Where do annotators disagree most?

Improving Agreement

Strategies:

1. **Better guidelines**:

- Define categories clearly
- Provide examples
- Handle edge cases

2. **Training**:

- Practice sessions
- Discuss disagreements
- Calibrate understanding

3. **Task simplification**:

Improving Agreement (continued)

4. **Quality control**:

- Gold standard examples
- Regular agreement checks
- Feedback to annotators

5. **Redundancy**:

- Multiple annotations per item
- Majority vote or expert resolution
- Identify difficult examples

Part 6: Best Practices

Building high-quality labeled datasets

Annotation Guidelines

Essential components:

1. **Task description**: What are we labeling?
2. **Label definitions**: Clear definition of each category
3. **Examples**: Show good and bad annotations
4. **Edge cases**: How to handle ambiguous cases
5. **Process**: Step-by-step instructions

Document everything!

Example Guidelines: Sentiment

Task: Label movie reviews as Positive, Negative, or Neutral

Definitions:

- **Positive**: Overall favorable opinion
- **Negative**: Overall unfavorable opinion
- **Neutral**: Mixed feelings or factual statement

Examples:

- "Amazing movie!" → Positive
- "Terrible waste of time" → Negative
- "The movie was 2 hours long" → Neutral

Edge cases:

Pilot Annotation

Before full-scale labeling:

1. **Small pilot** (10-20 examples)
2. **Multiple annotators** label same data
3. **Measure agreement**
4. **Identify issues**:
    - Unclear guidelines
    - Ambiguous categories
    - Missing edge cases
5. **Revise** and repeat

Don't skip this step!

Quality Control Strategies

1. **Gold standard**:

- Expert-labeled examples
- Test annotators periodically
- Track performance

2. **Redundancy**:

- 2-3 annotations per item
- Resolve disagreements
- Filter low-quality work

3. **Regular reviews**:

- Spot-check annotations

Handling Disagreements

When annotators disagree:

Option 1: Majority vote

```
Item 1: A=Positive, B=Positive, C=Negative
Label: Positive (2/3)
```

Option 2: Expert adjudication

- Expert reviews disagreements
- Makes final decision

Option 3: Discard

- If no clear answer, skip item
- Keep only high-confidence labels

Dataset Split Strategy

Standard splits:

Training (70-80%): Model learns from this

Validation (10-15%): Tune hyperparameters

Test (10-15%): Final evaluation

Important: Keep annotators consistent within splits!

Bad: Different annotators for train vs test

Good: Same quality standards throughout

Annotation Cost Estimation

Factors:

- Task complexity (simple classification vs detailed segmentation)
- Annotator expertise (crowd vs expert)
- Quality requirements (single vs multiple annotations)
- Dataset size (hundreds vs millions)

Example estimates:

- Image classification: $0.01 - $0.10 per image
- Bounding boxes: $0.10 - $1.00 per image
- Segmentation: $1.00 - $10.00 per image
- NER: $0.05 - $0.50 per sentence

Active Learning

Reduce annotation cost:

Strategy:

1. Label small initial dataset
2. Train initial model
3. **Model selects** most useful examples to label next
4. Label those examples
5. Retrain and repeat

Key idea: Label the data that helps the model most

Savings: Can reduce labeling by 50-90%

Common Pitfalls

1. **Vague guidelines**: Leads to disagreement
2. **Too many classes**: Hard to distinguish
3. **Imbalanced data**: Too few examples of some classes
4. **Batch effects**: Annotators change behavior over time
5. **No validation**: Don't know if labels are good
6. **Ignoring context**: Labels without understanding domain

Solution: Plan carefully, test early, iterate

Summary: Labeling Workflow

Complete process:

1. **Define task**: What are we predicting?
2. **Create guidelines**: Clear definitions + examples
3. **Pilot test**: Small batch, measure agreement
4. **Revise**: Fix issues identified
5. **Scale up**: Full annotation with quality control
6. **Validate**: Check agreement throughout
7. **Export**: ML-ready format
8. **Split**: Train/val/test sets

Result: High-quality labeled dataset for ML

Next Steps

Homework:

- Set up Label Studio
- Create annotation project for your Netflix data
- Annotate sample of movie posters or reviews
- Measure inter-annotator agreement

Next topics:

- Training models on labeled data
- Evaluation metrics
- Active learning strategies

Key Takeaways

1. **Labeling is critical**: Quality labels = quality models
2. **Many task types**: Classification, detection, NER, etc.
3. **Tools help**: Label Studio makes annotation easier
4. **Measure quality**: Use Kappa to check agreement
5. **Guidelines matter**: Clear instructions improve consistency
6. **Iterate**: Pilot → revise → scale up

Resources

Tools:

- Label Studio: https://labelstud.io/
- CVAT (computer vision): https://cvat.org/
- Prodigy (text): https://prodi.gy/

Metrics:

- scikit-learn metrics: https://scikit-learn.org/stable/modules/model_evaluation.html
- Agreement calculators

Reading:

- Best practices for data annotation
- Active learning tutorials

Questions?

Next class: Lab session - hands-on annotation with Label Studio