Annotation

approaches, best practices, and debates

Terminology

- **example/sample/item**: an observation (“row”) in your dataset (e.g., document, paragraph, or sentence)
- **label**: the category an example is assigned to
- **coder/annotator**: the agent (human or machine) that provides a judgment/opinion about which label should be assigned to a given example
- **coding/annotation**: the coder’s/annotator’s judgment

Approaches

how to label text data

- expert coders
- trained coders
- crowd coders
- human-in-the-loop
- LLMs

Types of coders

Two traditional types of coders

- experts
- trained coders

Innovations

- crowd workers (since ~2010)
- pre-trained classifiers or LLMs
  - zero-shot
  - in-context learning

Types of coders

Two traditional types of coders

- experts
- trained coders

Innovations

- crowd workers (since ~2010)
- pre-trained classifiers or LLMs
  - zero-shot
  - in-context learning

Literature

- see Krippendorff "Content Analysis: An Introduction to Its Methodology" on experts and trained coders
- see Benoit *et al.* (2016) for optimistic view on crowd coding
  - more opinions: here, here, here
- research on LLMs for annotation: here, here, here, here, and here (but still *many* open meth. questions)

Types of coders

Two traditional types of coders

- **experts**
- trained coders

Innovations

- crowd workers (since ~2010)
- pre-trained classifiers or LLMs
  - zero-shot
  - in-context learning

Experts

- assumed to know the target concept
- assumed to be able to reliably detect/classify the concept in texts
- *pros:* nice if assumptions hold
- *cons:*
  - subjectivity ⇒ reliability and replicability?
  - expensive

Types of coders

Two traditional types of coders

- experts
- **trained coders**

Innovations

- crowd workers (since ~2010)
- pre-trained classifiers or LLMs
  - zero-shot
  - in-context learning

Trained coders

- assumed to be able to learn reliably detecting the concept in texts
- *pros:*
  - reliability can be estimated
  - more replicable
- *cons:*
  - coders need to be trained
    ⇒ time-consuming
  - expensive
  - achieving reliability can be difficult for hard concepts

Types of coders

Two traditional types of coders

- experts
- trained coders

Innovations

- **crowd workers** (since ~2010)
- pre-trained classifiers or LLMs
  - zero-shot
  - in-context learning

Trained coders

- assumed to be motivated and easily learn to reliably detect the concept
- *pros:*
  - performance can be checked
  - reliability can be estimated
  - more replicable
- *cons:*
  - coders are paid poorly
  - little time to learn and code
  - achieving reliability can be difficult with any concept

Types of coders

Two traditional types of coders

- experts
- trained coders

Innovations

- **crowd workers** (since ~2010)
- pre-trained classifiers or LLMs
  - zero-shot
  - in-context learning

Crowd workers

- need to be taught the target concept
- assumed to be able to learn reliably detecting the concept in texts
- *pros:*
  - reliability can be estimated
  - more replicable (if documented)
- *cons:*
  - expensive (bc paid coders)
  - time-consuming (bc training)

Types of coders

Two traditional types of coders

- experts
- trained coders

Innovations

- crowd workers (since ~2010)
- **pre-trained classifiers or LLMs**
  - zero-shot
  - in-context learning

Pre-trained classifiers

- assumed to be able to transfer well from fine-tuning to target corpus
- *pros:*
  - zero-shot classification
  - cheap: labeled data only for test data
- *cons:*
  - out-of-domain classification
  - unlcear best practices in CSS

Types of coders

Two traditional types of coders

- experts
- trained coders

Innovations

- crowd workers (since ~2010)
- **pre-trained classifiers or LLMs**
  - zero-shot
  - in-context learning

LLMs

- assumed to have language understanding and reasoning abilities needed to translate prompt into judgments
- *pros:*
  - zero-shot classification and in-context learning
  - cheap: labeled data only for test data
- *cons:*
  - replicability
  - unclear best practices in CSS

Human-in-the-loop a.k.a *active learning*

Idea train classifier and gather annotations for hard examples to iteratively improve it

- start with small sample of annotated examples (e.g., using dictionary, zero-shot classifier, or in-context learning) and train classifier
- predict labels for unannoted expls.
- select most uncertain examples and get annotation from human coder
- update classifier with new annotations

Literature (Pol Sci)

- Hillard *et al.* (2008)
- Wiedemann (2019)
- Miller *et al.* (2021)
- Dai & Kustov (2022) *applied paper*
- Bosley *et al.* (2023)

in Python

`small-text` + **`argilla`** + `transformers` (see here)

Best practices

what to do and what to avoid

- concept development
- codebooks & instructions
- coder training
- quality assurance & assessment
- aggregating judgments

Developing concepts for text annotation

When you have a concept in mind you want to capture via text annotation …

- think about what's the best level of annotation (document, paragraph, sentence, word)

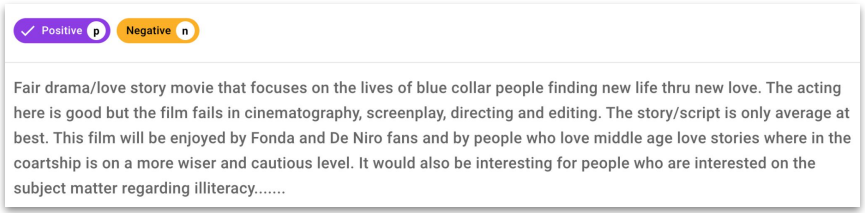
Level of annotation

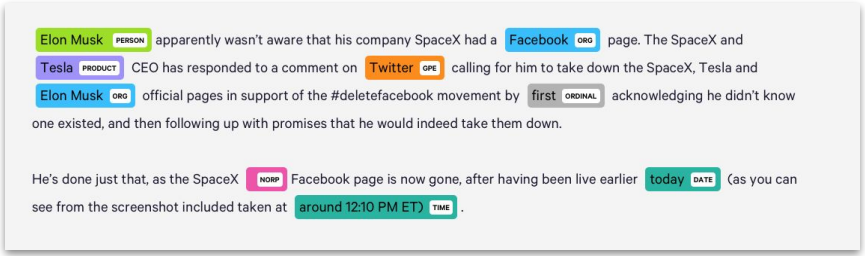
- **document** ⇒ "holistic grading" (see here, for example)
- **paragraph** ⇒ sequence classification (1+ label per para.)
- **sentence** ⇒ sequence classification (1+ label per sent.)
- **pairs** of sentences (see here)
- **word** ⇒ "token classification" (1 label per word, see here)

Developing concepts for text annotation

When you have a concept in mind you want to capture via text annotation …

- think about what's the best level of annotation (document, paragraph, sentence, word)

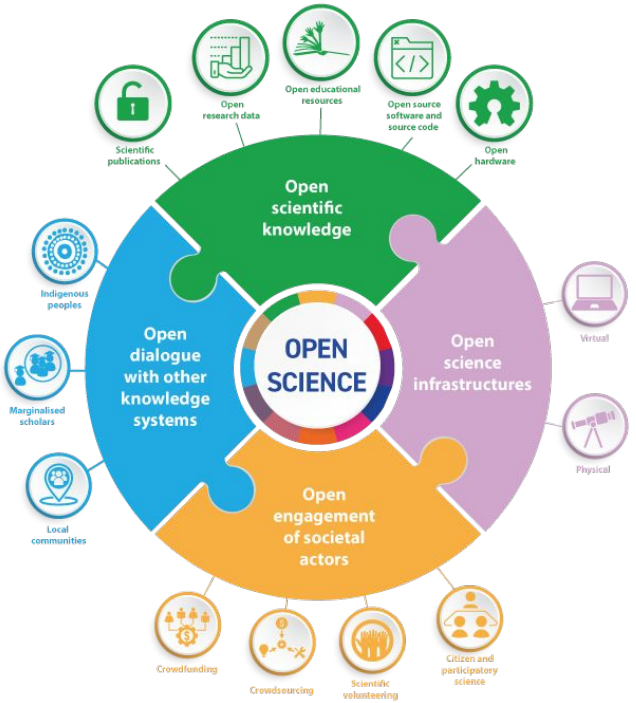




Codebook and instructions

Always document and publish the codebook and coding instructions!

- **codebook**
  - define concept
  - name label classes
  - describe text materials
- **coding instructions**
  - describe label classes
  - give good examples
  - give instructions what to do with hard and boundary cases



Coder training

Always sit down for at least one round of feedback with your coders!

- discuss easy cases where they misclassified
- discuss boundary cases they got wrong

For crowd coding: use “screening tasks”

Coder training

Always sit down for at least one round of feedback with your coders!

- discuss easy cases where they misclassified
- discuss boundary cases they got wrong

For crowd coding: use “screening tasks”

For crowd coding

Usually no direct communication with coders possible ⇒ no training iterations

- use only for not-to-hard concepts
- provide concise task description
  - use simple language (no jargon)
  - use a few good examples
- use “screening tasks” to select reliable coders at beginning
- always collect 3+ codings/example

Collecting annotations

Three guiding principles

- Labeling should be coder-friendly
  - usability ⇒ little frustration
  - the less frustration ⇒ the better the annotations
- Think about the ideal final data format before deciding for a tool

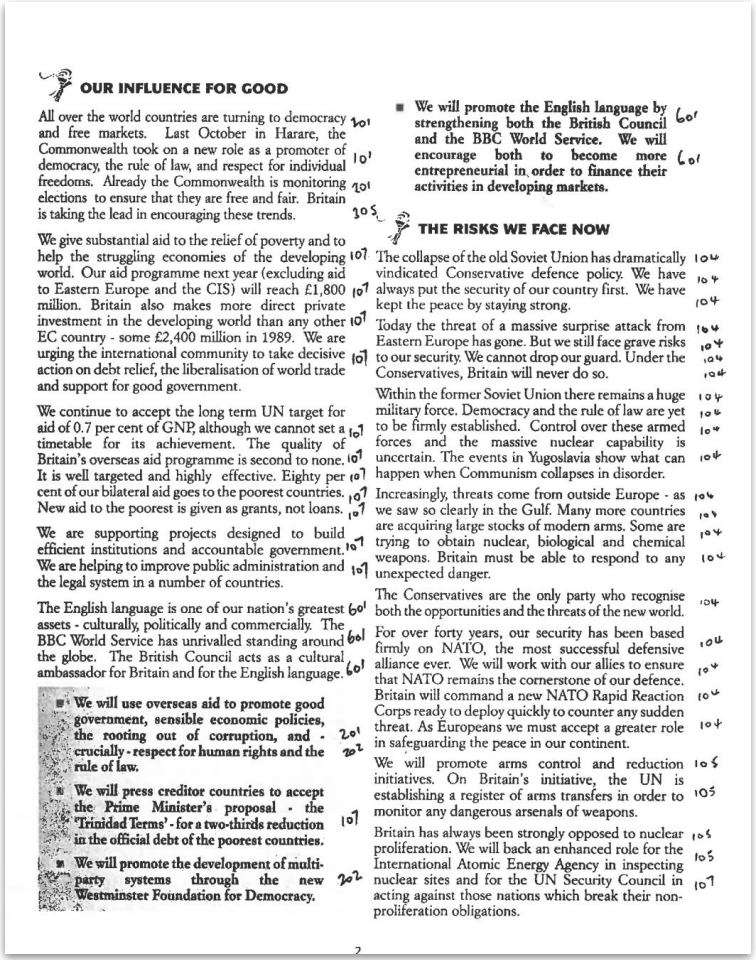
Options

- just put sentences in a Google Sheets file with an extra column for judgments; ask coders to make a copy and share a link with you
- for word-level annotation: use a word file (and parse it with python)
- use an App (doccano or argilla)

Collecting annotations

Three guiding principles

- Labeling should be coder-friendly
  - usability ⇒ little frustration
  - the less frustration ⇒ the better the annotations
- Think about the ideal final data format before deciding for a tool
- The annotation process should be reproducible ⇒ use ***digital*** tools



OUR INFLUENCE FOR GOOD

All over the world countries are turning to democracy and free markets. Last October in Harare, the Commonwealth took on a new role as a promoter of democracy, the rule of law, and respect for individual freedoms. Already the Commonwealth is monitoring elections to ensure that they are free and fair. Britain is taking the lead in encouraging these trends.

We give substantial aid to the relief of poverty and to help the struggling economies of the developing world. Our aid programme next year (excluding aid to Eastern Europe and the CIS) will reach £1,800 million. Britain also makes more direct private investment in the developing world than any other EC country - some £2,400 million in 1989. We are urging the international community to take decisive action on debt relief, the liberalisation of world trade and support for good government.

We continue to accept the long term UN target for aid of 0.7 per cent of GNP, although we cannot set a timetable for its achievement. The quality of Britain's overseas aid programme is second to none. It is well targeted and highly effective. Eighty per cent of our bilateral aid goes to the poorest countries. New aid to the poorest is given as grants, not loans.

We are supporting projects designed to build efficient institutions and accountable government. We are helping to improve public administration and the legal system in a number of countries.

The English language is one of our nation's greatest assets - culturally, politically and commercially. The BBC World Service has unrivalled standing around the globe. The British Council acts as a cultural ambassador for Britain and for the English language.

- **We will use overseas aid to promote good government, sensible economic policies, the rooting out of corruption, and - crucially - respect for human rights and the rule of law.**
- **We will press creditor countries to accept the Prime Minister's proposal - the 'Trinidad Terms' - for a two-thirds reduction in the official debt of the poorest countries.**
- **We will promote the development of multi-party systems through the new Westminster Foundation for Democracy.**
- **We will promote the English language by strengthening both the British Council and the BBC World Service. We will encourage both to become more entrepreneurial in order to finance their activities in developing markets.**

THE RISKS WE FACE NOW

The collapse of the old Soviet Union has dramatically vindicated Conservative defence policy. We have always put the security of our country first. We have kept the peace by staying strong.

Today the threat of a massive surprise attack from Eastern Europe has gone. But we still face grave risks to our security. We cannot drop our guard. Under the Conservatives, Britain will never do so.

Within the former Soviet Union there remains a huge military force. Democracy and the rule of law are yet to be firmly established. Control over these armed forces and the massive nuclear capability is uncertain. The events in Yugoslavia show what can happen when Communism collapses in disorder.

Increasingly, threats come from outside Europe - as we saw so clearly in the Gulf. Many more countries are acquiring large stocks of modern arms. Some are trying to obtain nuclear, biological and chemical weapons. Britain must be able to respond to any unexpected danger.

The Conservatives are the only party who recognise both the opportunities and the threats of the new world.

For over forty years, our security has been based firmly on NATO, the most successful defensive alliance ever. We will work with our allies to ensure that NATO remains the cornerstone of our defence. Britain will command a new NATO Rapid Reaction Corps ready to deploy quickly to counter any sudden threat. As Europeans we must accept a greater role in safeguarding the peace in our continent.

We will promote arms control and reduction initiatives. On Britain's initiative, the UN is establishing a register of arms transfers in order to monitor any dangerous arsenals of weapons.

Britain has always been strongly opposed to nuclear proliferation. We will back an enhanced role for the International Atomic Energy Agency in inspecting nuclear sites and for the UN Security Council in acting against those nations which break their non-proliferation obligations.

Quality assurance & assessment

Annotation quality

- important for supervised learning
  - bad annotation result in “noisy” labels
  - noisy labels impair ability to learn the relevant signal
- related to replicability: if coders can agree, task should be replicable
- commonly quantified with inter-coder reliability metrics

Quality assurance & assessment

Annotation quality

- important for supervised learning
  - bad annotation result in “noisy” labels
  - noisy labels impair ability to learn the relevant signal
- related to replicability: if coders can agree, task should be replicable
- commonly quantified with inter-coder reliability metrics

Quality assurance

- train coders
- always collect 2+ codings/example if feasible possible (see here)
  - training good classifiers requires less and les labeled data
  - use resources to invest in quality, not quantity !!!
- review disagreement case

Quality assurance & assessment

Annotation quality

- important for supervised learning
  - bad annotation result in “noisy” labels
  - noisy labels impair ability to learn the relevant signal
- related to replicability: if coders can agree, task should be replicable
- commonly quantified with inter-coder reliability metrics

Inter-coder reliability

- just % agreement is *not* enough (need to adjust for baseline)
- compute “chance-adjusted” agreement metrics
  - Krippendorff’s *alpha*
  - Cohen’s *kappa*

read here and here

https://github.com/Toloka/crowd-kit

Aggregating judgments

If you collect 2+ judgments per example, you need to aggregate them at the example level

Approaches

- plurality voting (default in CSS)

Aggregating judgments

If you collect 2+ judgments per example, you need to aggregate them at the example level

Approaches

- plurality voting (default in CSS)
- (Bayesian) annotation aggregation, see here and here: estimate latent labels from annotation
  - un- or semi-supervised
  - can account for variation in coders' (unobserved) abilities

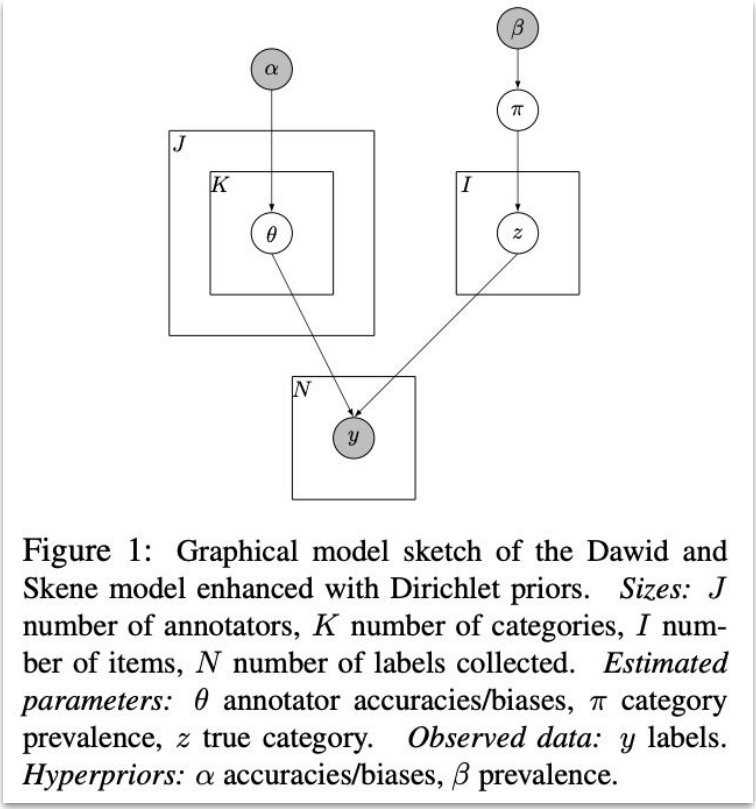


Figure 1: Graphical model sketch of the Dawid and Skene model enhanced with Dirichlet priors. *Sizes:* J number of annotators, K number of categories, I number of items, N number of labels collected. *Estimated parameters:* θ annotator accuracies/biases, π category prevalence, z true category. *Observed data:* y labels. *Hyperpriors:* α accuracies/biases, β prevalence.

Aggregating judgments

If you collect 2+ judgments per example, you need to aggregate them at the example level

Approaches

- plurality voting (default in CSS)
- (Bayesian) annotation aggregation, see here and here: estimate latent labels from annotation
  - un- or semi-supervised
  - can account for variation in coders' (unobserved) abilities

But I believe you could also directly train on annotations with built-in mechanisms to aggregate (CS literature)

- Rodrigues and Perreira (2018)
- Matin & Valdenegro-Toro (2020)

Debates

open questions and what’s next

- best practices for LLMs?
- LLMs or fine-tuning?

Debates

LLM best practices?

- open- *vs.* closed-source models
- accessibility ⇒ equity and representation
- prompt engineering strategies and techniques are more of an art than a science currently

In-context learning with LLMs or fine-tuning?

- fine-tuning requires more data but better replicable and more accessible (smaller models)
- LLMs are more powerful

Debates

LLM best practices?

- open- *vs.* closed-source models
- accessibility ⇒ equity and representation
- prompt engineering strategies and techniques are more of an art than a science currently

In-context learning with LLMs or fine-tuning?

- fine-tuning requires more data but better replicable and more accessible (smaller models)
- LLMs are more powerful