Toloka

Toloka

Crowdsourcing for Information Retrieval

Tutorial at ECIR ‘23

Dmitry Ustalov, Natalia Fedorova, Nikita Pavlichenko,
Alisa Smirnova, Daniil Likhobaba

Toloka

Part IV
Learning from Crowds

Dr. Dmitry Ustalov,
Head of Ecosystem Development Unit at Toloka

Tutorial Schedule

Part I Intro: 15 min
Introduction

Part II: 45 min
Crowdsourcing Essentials

Part III: 30 min
Hands-On Practice Session

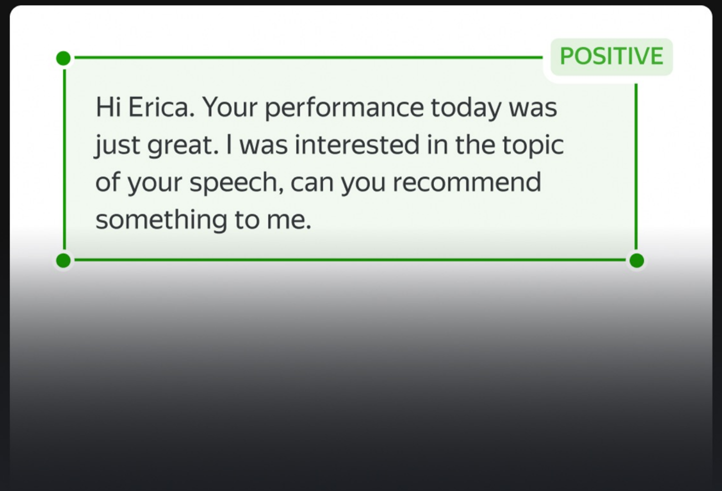
Coffee Break :
20 min

Part III: 30 min
Hands-On Practice Session

Part IV: 45 min
Learning from Crowds

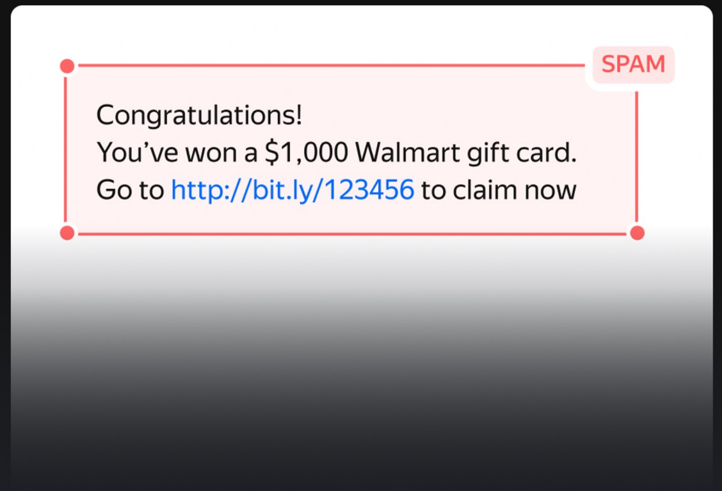
Part V: 15 min
Conclusion

Use Cases



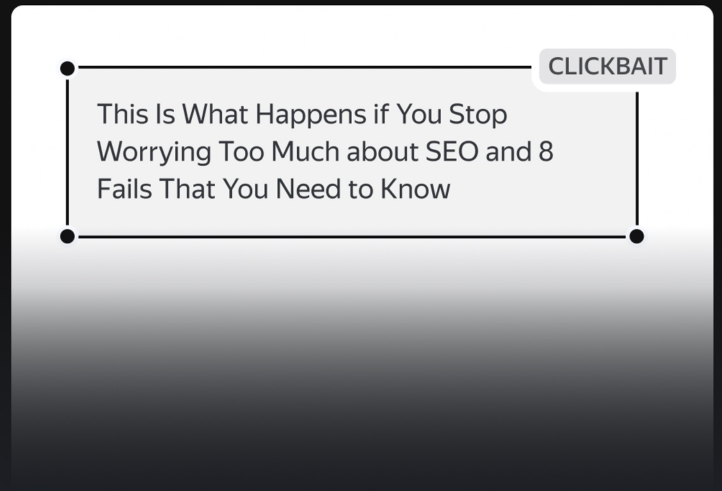
Sentiment Analysis

Classifies text content into 3 classes — positive, negative, and neutral.



Spam Detection

Handles classic spam content classification. Easily tuned to your data streams.



Text Moderation

Detects problematic content like spam, clickbait, hate speech, and profanity.

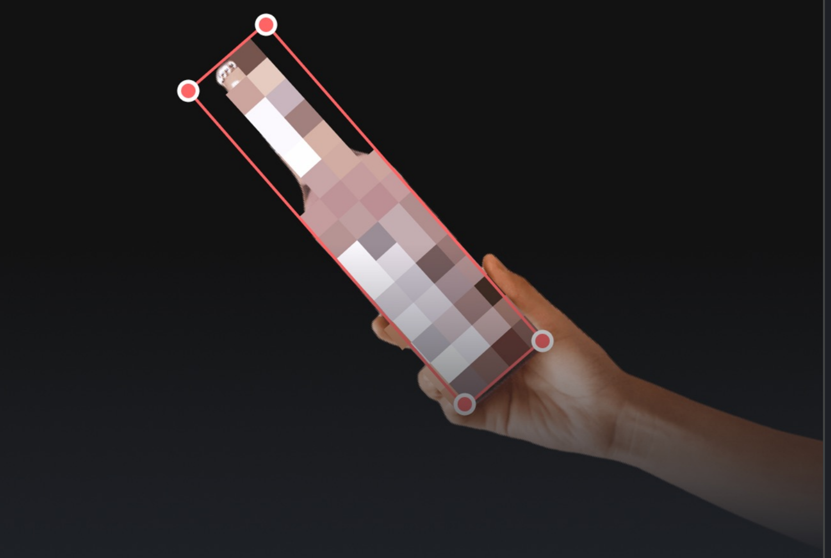
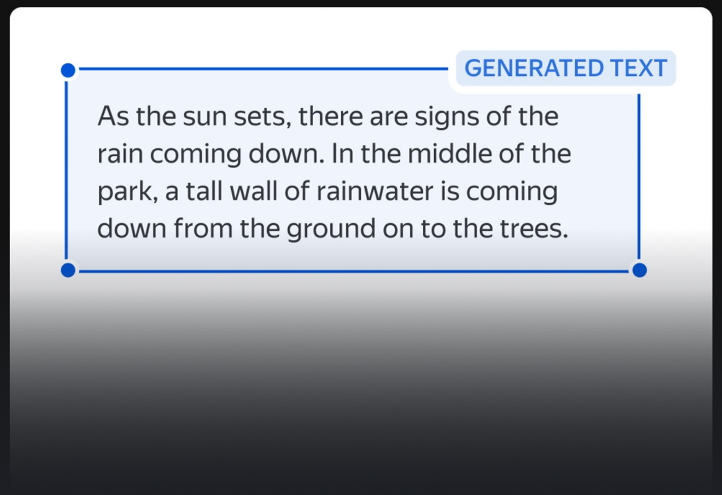


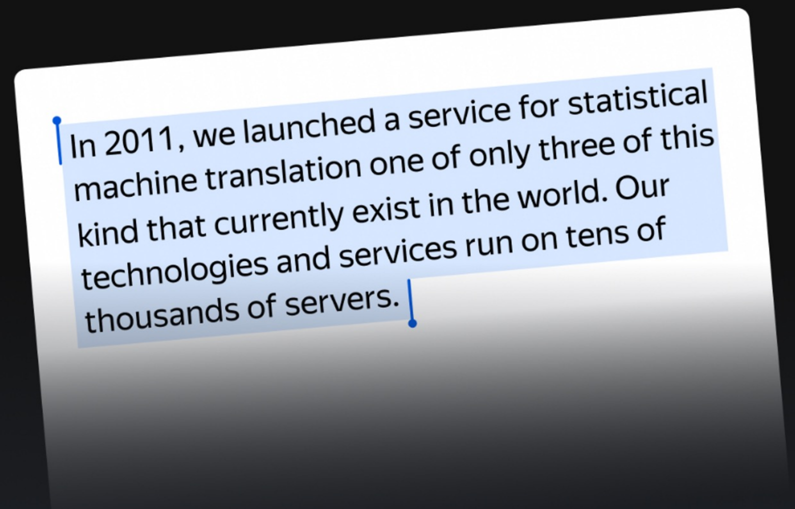
Image Moderation

Detects adult content, illegal content, copyright infringement, and other problematic images.



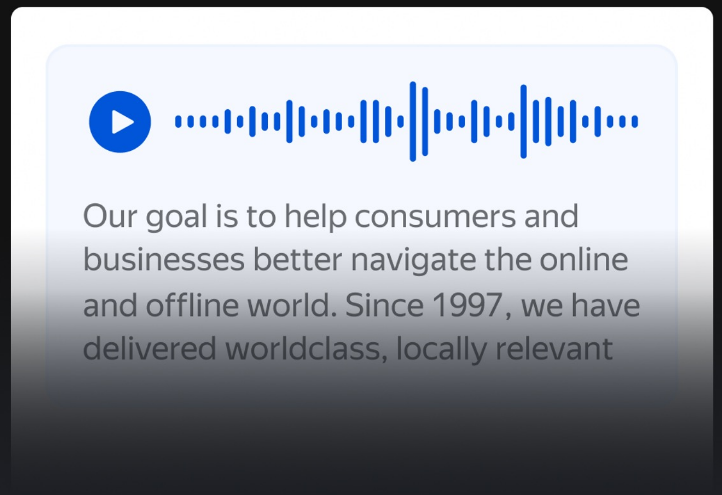
Multilingual Large Transformer

GPT-3-like model classifies and generates short texts in 12 languages.



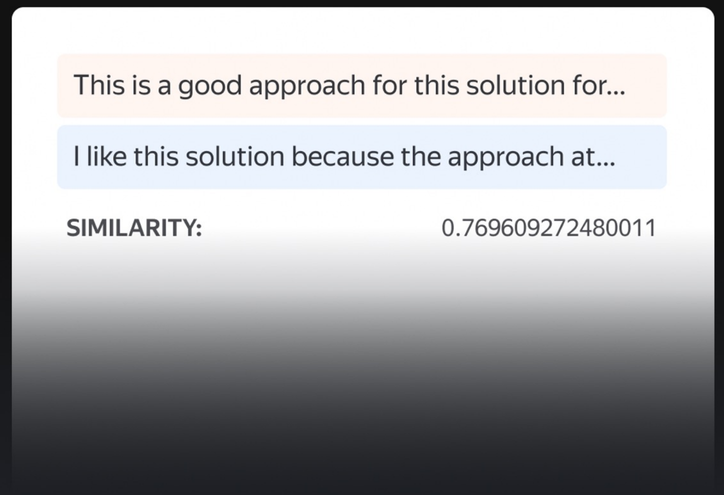
Optical Character Recognition (OCR)

Extracts text from images in more than 40 languages.



Speech-to-Text

Captures text from audio content in 13 different languages.



Semantic Similarity

Compares 2 texts based on similarity in meaning.

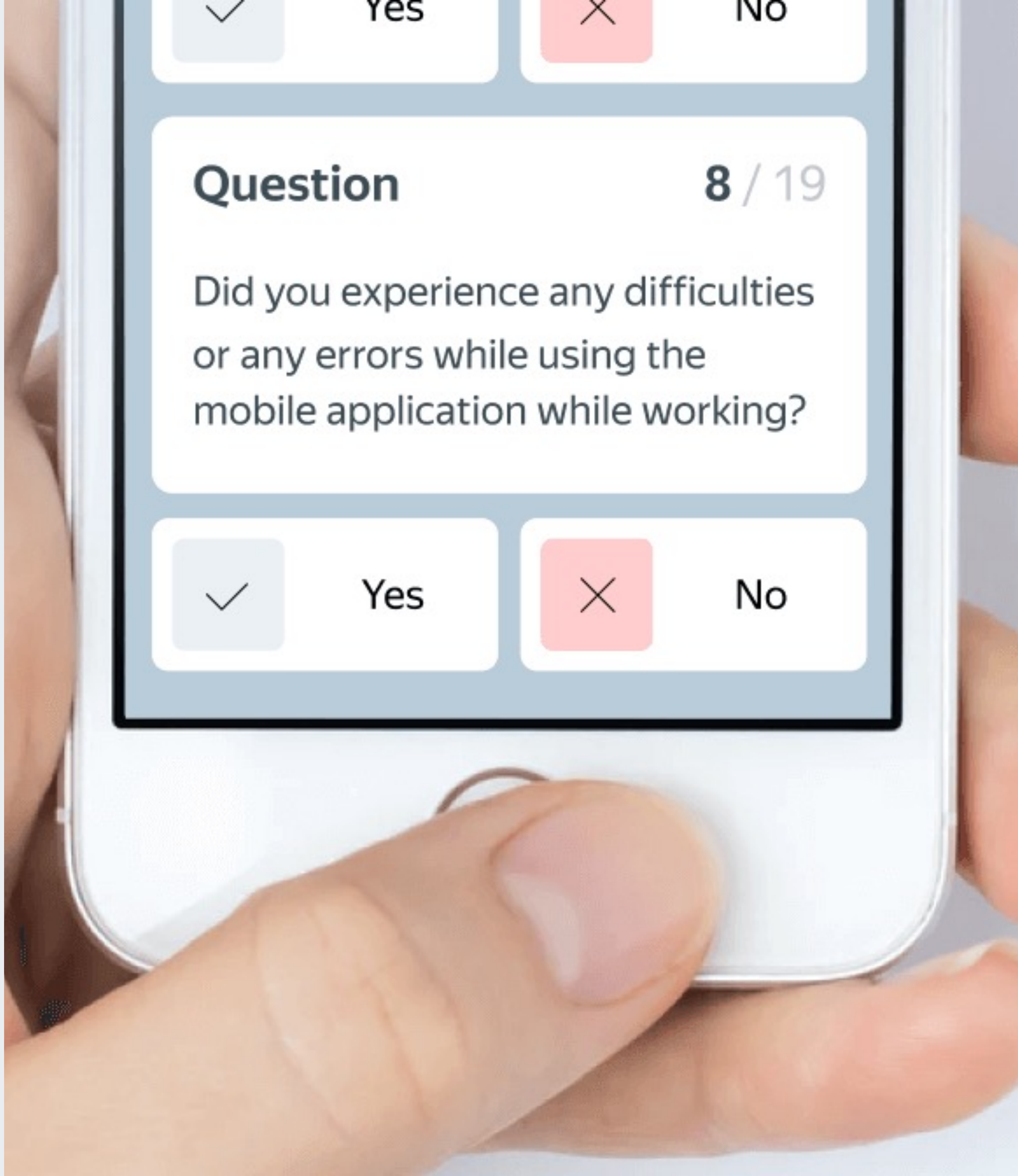
… and more!

Do You Trust Your Labels?

It is often true that there is only one correct label per task, but

- crowd annotators are not experts in your task or domain
- experts make mistakes, too
- user-generated content might be fuzzy

This issue is solved using *consensus* by asking **multiple different people**.



It’s great if we have multiple labels per object.

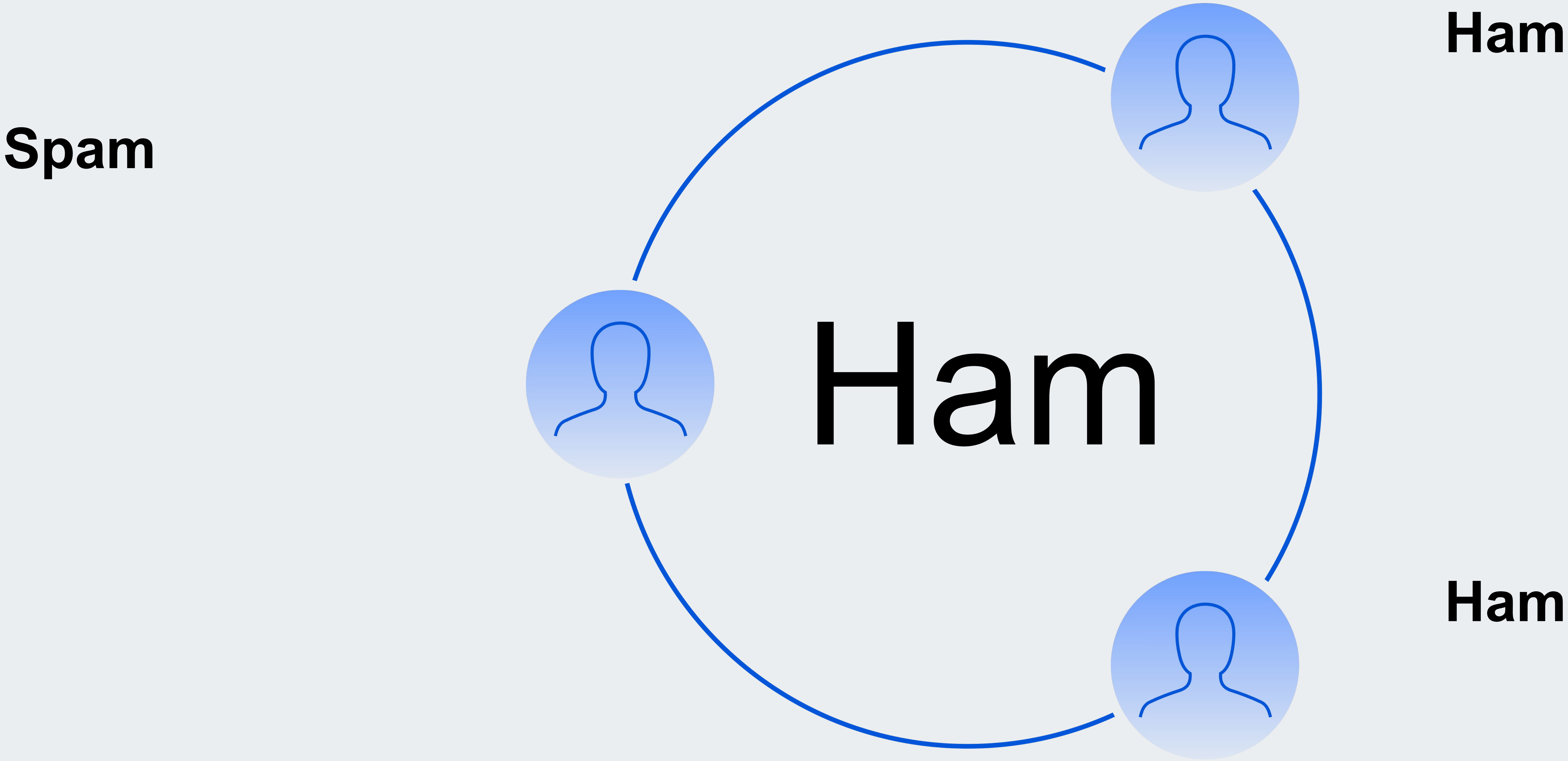
…but now we need to do something with these extra labels!

How to Select the Label?

There is an obvious (but not always correct) way

How to Select the Label?

There is an obvious (but not always correct) way



It is called **consensus**, **aggregation**, or **truth inference** problem.

How Good is Majority Vote (MV)?

Method	D_Product	D_PosSent	S_Rel	S_Adult	binary1	binary2
MV	0.897	0.932	0.536	0.763	0.931	0.936
Wawa	0.897	0.951	0.557	0.766	0.981	0.983
DS	0.940	0.960	0.615	0.748	0.994	0.994
GLAD	0.928	0.948	0.511	0.760	0.994	0.994
KOS	0.895	0.933	—	—	0.993	0.994
MACE	0.929	0.950	0.501	0.763	0.995	0.995
M-MSR	—	0.937	0.425	0.751	0.994	0.994

Table 3: Comparison of the implemented categorical aggregation methods (accuracy is used).

Truth Inference Models

Majority Vote (MV)

labels are similar

tasks are similar

annotators are similar

Wawa

labels are similar

tasks are similar

annotators are different

Dawid-Skene (DS)

labels are different

tasks are similar

annotators are different

Dawid-Skene (1979), an EM algorithm

The algorithm is initialized with MV; notation y_j^w means the label received from annotator w for task j

E step for true labels ($\hat{z}_j$):

$$\hat{z}_j[c] = \frac{p[c] \prod_{w \in W_j} e^w[c, y_j^w]}{\sum_k p[k] \prod_{w \in W_j} e^w[k, y_j^w]}, \qquad c = 1, \dots, K$$

M step for error matrices of annotators (e^w):

$$e^w[c,k] = \frac{\sum_{j \le J} \hat{z}_j[c] \delta(y_j^w = k)}{\sum_{q=1}^K \sum_{j \le J} \hat{z}_j[c] \delta(y_j^w = q)}, \qquad k, c = 1, \dots, K$$

M step for label priors (p):

$$p[c] = \frac{\sum_{j \le J} \hat{z}_j[c]}{J}, \qquad c = 1, \dots, K$$

Example: Dawid-Skene (1979)

Task \ Annotator	w_1	w_2	w_3	w_4	w_5
t_1	ham	spam		spam	ham
t_2	spam	spam	spam	ham	ham
t_3	spam	ham	ham	spam	ham
t_4	spam	spam	spam	spam	spam
t_5	spam	ham	ham	ham	ham

Example: Dawid-Skene (1979)

Task \ Annotator	w_1	w_2	w_3	w_4	w_5
t_1	ham	spam		spam	ham
t_2	spam	spam	spam	ham	ham
t_3	spam	ham	ham	spam	ham
t_4	spam	spam	spam	spam	spam
t_5	spam	ham	ham	ham	ham

Task \ Label	ham	spam
t_1	0.15	0.85
t_2	0.10	0.90
t_3	0.99	0.01
t_4	0.00	1.00
t_5	1.00	0.00

Some datasets, like **ImageNet**, include only aggregated labels, not raw labels.

The problem is way too popular, there are so many methods (Zheng et al., VLDB ’17).

One needs to choose the model using the held-out dataset.

However, it is sufficient to use **MV** on smaller datasets and **DS** on larger datasets.

…but aggregation does not take into account the task content.

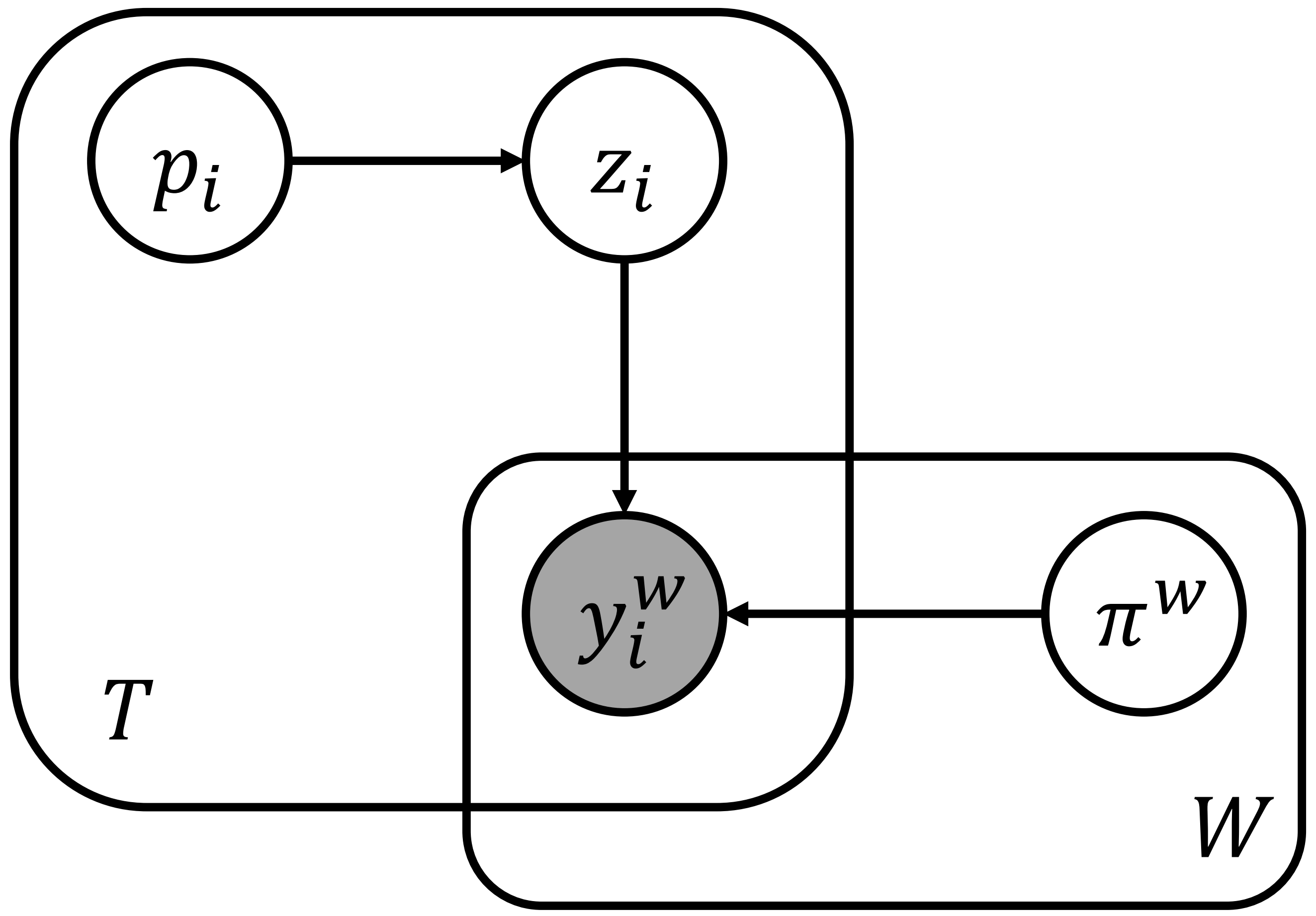
…but aggregation does not take into account the task content.

What do we do?

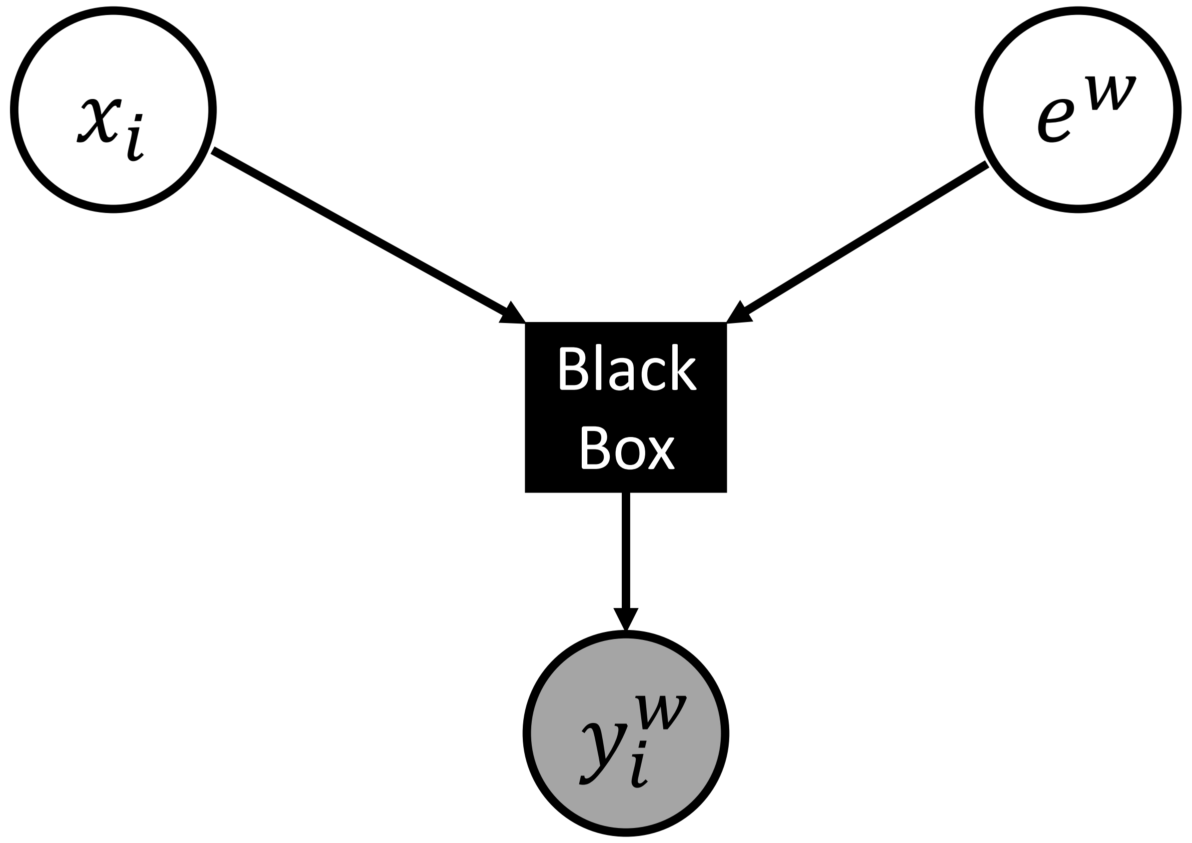
Suppose that our **input** is a text,
an image, or a video,
and the **output** is the class label.

Avoiding the Aggregation Step

It is possible to train (or fine-tune) the model using the raw labels without aggregation!



vs.



Deep Learning from Crowds

We usually *train* or *fine-tune* the pre-trained backbone model that transforms our object as a vector $\boldsymbol{x}$, so our **classification function** is

$$\mathrm{MLP}\big(\mathrm{Backbone}(\boldsymbol{x})\big)$$

However, if we will train only on the responses, we will lose important information about *annotators* and *tasks*!

Can We Do Better?

CrowdLayer (Rodrigues & Pereira, AAAI '18) is a method that learns the confusion matrix $\boldsymbol{A}_w$ of every annotator w.

The classification function becomes

$$\boldsymbol{A}_w \mathrm{MLP}\big(\mathrm{Backbone}(\boldsymbol{x})\big)$$

Also, there are more complex methods:
SpeeLFC (Chen et al., IJCAI '20),
CoNAL (Chu et al., AAAI '21), etc.

Can We Do Better?

CoNAL (Chu et al., AAAI ‘21) is a method that learns annotator-specific confusion matrices $\boldsymbol{A}_w$ and one common confusion matrix $\boldsymbol{A}_g$.

The resulting prediction is a blending of two confusions where blending coefficient is a scalar product of task $\boldsymbol{x}_t$ and annotator features $\boldsymbol{x}_w$.

The classification function is

$$\alpha \boldsymbol{A}_w \mathrm{MLP}\big(\mathrm{Backbone}(\boldsymbol{x}_t)\big) + (1-\alpha)\boldsymbol{A}_g \mathrm{MLP}\big(\mathrm{Backbone}(\boldsymbol{x}_t)\big)$$

$$\alpha = \mathrm{sigmoid}(\boldsymbol{x}_t \cdot \boldsymbol{x}_w)$$

It Works!

Dataset	Backbone	CoNAL	CrowdLayer	Base	DS	MV
IMDb	LSTM	$\boxed{0.844}$	0.825	0.835	0.841	0.819
IMDb	RoBERTa	$\boxed{0.932}$	0.928	0.927	$\boxed{0.932}$	0.927
CIFAR-10	VGG-16	0.825	0.863	$\boxed{0.882}$	0.877	0.865

Table 7: Comparison of different methods for deep learning from crowds with traditional answer aggregation methods (test set accuracy is used).

Training on raw labels allows to skip the aggregation step, **but it loses information about the annotators**.

This approach works only **if we can represent our object as a vector** (so almost always).

There are specialized models that incorporate annotator information to increase the prediction quality.

There are specialized models that incorporate annotator information to increase the prediction quality.

Be careful about the assumptions!

…but I don’t want any formulas!

Fair enough.

Crowd-Kit

Crowd-Kit is a Python library that implements popular quality control techniques for crowdsourcing:

— answer aggregation and learning from crowds

— quality and inter-annotator agreement metrics

— dataset loaders and transformers

https://github.com/Toloka/crowd-kit (Apache License 2.0)

Dawid-Skene Aggregation

See more at https://github.com/Toloka/crowd-kit/blob/main/examples/TlkAgg-Categorical.ipynb

```
from crowdkit.datasets import load_dataset
from crowdkit.aggregation import DawidSkene

# df is a pd.DataFrame with categorical responses
# gt is a pd.Series with ground truth answers
df, gt = load_dataset('relevance-2')

# ds is a Crowd-Kit implementation of the Dawid-Skene model
ds = DawidSkene(n_iter=10)

# agg is a pd.Series with objects and their categories
agg = ds.fit_predict(df)
```

Crowd-Kit Evaluation

Method	D_Product	D_PosSent	S_Rel	S_Adult	binary1	binary2
MV	0.897	0.932	0.536	0.763	0.931	0.936
Wawa	0.897	0.951	0.557	0.766	0.981	0.983
DS	0.940	0.960	0.615	0.748	0.994	0.994
GLAD	0.928	0.948	0.511	0.760	0.994	0.994
KOS	0.895	0.933	—	—	0.993	0.994
MACE	0.929	0.950	0.501	0.763	0.995	0.995
M-MSR	—	0.937	0.425	0.751	0.994	0.994

Table 3: Comparison of the implemented categorical aggregation methods (accuracy is used).

Dataset	Version	ROVER	RASA	HRRASA
CrowdWSA	J1	0.612	0.659	0.676
	T1	0.514	0.483	0.500
	T2	0.524	0.498	0.520
CrowdSpeech	dev-clean	0.676	0.750	0.745
	dev-other	0.132	0.142	0.142
	test-clean	0.729	0.860	0.859
	test-other	0.134	0.157	0.157

Table 5: Comparison of implemented sequence aggregation methods (average word error rate is used).

Method	Chen et al. (2013)	IMDB-WIKI-SBS
Bradley-Terry	0.246	0.737
noisyBT	0.238	0.744
Random	-0.013	-0.001

Table 4: Comparison of implemented pairwise aggregation methods (Spearman's ρ is used).

Dataset	MV	EM	RASA
MS COCO	0.839	0.861	0.849

Table 6: Comparison of implemented image aggregation algorithms (IoU is used).

Dataset	Backbone	CoNAL	CrowdLayer	Base	DS	MV
IMDb	LSTM	0.844	0.825	0.835	0.841	0.819
IMDb	RoBERTa	0.932	0.928	0.927	0.932	0.927
CIFAR-10	VGG-16	0.825	0.863	0.882	0.877	0.865

Table 7: Comparison of different methods for deep learning from crowds with traditional answer aggregation methods (test set accuracy is used).

References

1. Zheng et al. "**Truth Inference in Crowdsourcing: Is the Problem Solved?**" *Proceedings of the VLDB Endowment*, vol. 10, no. 5, Jan. 2017, pp. 541–552. https://doi.org/10.14778/3055540.3055547
2. Uma et al. "**Learning from Disagreement: A Survey**." *Journal of Artificial Intelligence Research*, vol. 72, Dec. 2021, pp. 1385–1470. https://doi.org/10.1613/jair.1.12752
3. Dawid and Skene. "**Maximum Likelihood Estimation of Observer Error-Rates Using the EM Algorithm**." *Applied Statistics*, vol. 28, no. 1, 1979, p. 20–28. https://doi.org/10.2307/2346806
4. Ustalov et al. "**Learning from Crowds with Crowd-Kit**." *arXiv*. https://arxiv.org/abs/2109.08584

Dmitry Ustalov

Head of Ecosystem
Development Unit

dustalov@toloka.ai

www.toloka.ai | LinkedIn | GitHub | slack

Join our
Slack community