National Research University Higher School of Economics
Faculty of Computer Science
Programmer ‘Master of Data Science’

MASTER’S THESIS

Review of noisy labels detection methods for image and text data

Student: Pochukaev Sergey Anatolievich

Supervisor: MS Eldar Ganbarov

Moscow, 2021

Обзор методов обнаружения шумной разметки для текстовых и картиночных данных

Почукаев Сергей Анатольевич

Представлено в
Национальный исследовательский университет
"Высшая школа экономики"
Декабрь 27, 2021

Аннотация

В работе анализируется возможность использования методов глубокого обучения в задачах обучения моделей на картиночных и текстовых данных с зашумленными метками. Поскольку зашумленные метки сильно снижают качество глубоких нейронных сетей, развитие методов работы с такими меткам становится важной задачей развития глубокого обучения.

С ростом сложности и уникальности задач, которые пытаются решить с помощью нейронных сетей, многие задачи больше не могут быть решены с использованием экспертных наборов данных, поэтому возрастает роль датасетов, собранных открытыми, краудсорсинговыми платформами (Яндекс Толока, Amazon Turk, Google Task Mate), для которых по многим причинам характерна проблема зашумления меток.

В работе описаны основные методы работы с зашумленными метками, представленные в научных статьях на данный момент. Также были выбраны и протестированы несколько методов работы с зашумленными метками (mixup, co-teaching, confident learning) на примерах реальных, размеченных в Яндекс Толоке датасетов изображений и текстов.

Ключевые слова: шумная разметка, глубокое обучение, Яндекс Толока, классификация

Review of noisy labels detection methods for image and text data

Pochukaev Sergey Anatolevich

Submitted to
Higher School of Economics
on December 27, 2021

Abstract

The paper analyzes the possibility of using deep learning methods in problems of training image and text models on datasets with noisy labels. As noisy labels severely degrade the generalization of deep neural networks, learning from noisy labels (robust training) is becoming an important task in modern deep learning applications.

With the increasing complexity and uniqueness of tasks that are trying to be solved using neural networks, many tasks can no longer be solved using traditional «collected by experts and clean» datasets, so the role and usages of datasets, collected by open platforms for crowdsourcing markup data collection (Yandex Toloka, Amazon Turk, Google Task Mate), are increasing, which are characterized by the problem of noisy labels for many reasons.

The paper describes the main methods of working with noisy labels presented in scientific papers at the moment. Also, several methods of working with noisy labels (distillation approach, co-teaching, confident learning) were selected and tested on the examples of real, marked-up by Yandex Toloka picture and text datasets.

Keywords: deep learning, noisy labels, label noise, robust training, robust deep learning, classification, Yandex Toloka

Table of contents

Chapter 1

Introduction

Deep learning has achieved remarkable success in numerous domains (images, texts, video and etc.) with the help of large amounts of big data. However, with the increase of the examined problems and neural network architecture complexity the availability of datasets for training is becoming bigger concern because of the lack of high-quality dataset labels in many real problems, which can only be collected by big tech companies or universities.

The answer for this problem was the creation of open, crowdsourcing platforms for markup data collection (Yandex Toloka, Amazon Turk, Google Task Mate), where companies and individual researchers can order a markup for a collected dataset, but these platforms have a drawback, that the labels, which were given by the workers on these platforms, are noisy, which mean that they can be wrong.

According to recent studies ([1], [5] and [6]), this label noise significantly influences the performance of deep learning models as noisy labels severely degrade the generalization ability of deep neural networks.

The paper describes the main methods of working with noisy labels presented in scientific papers at the moment. Also, several methods of working with noisy labels (mixup, co-teaching, confident learning) were selected and tested on the examples of real, marked-up by Yandex Toloka picture and text datasets.

These methods were tested on real image and text datasets, which were marked in Russian crowdsourced service, Yandex Toloka. Also, the advantages and drawbacks of the chosen methods are presented after methods comparison.

1. Problem setup

The concept of label noise is ambiguous and is not easy to define. In scientific papers label noise usually presented as a mislabeled or corrupted instances in the dataset. As this paper will use datasets from open crowdsourced platforms, such as Yandex Toloka, Amazon Turk, etc., these label imperfections can be the result of several problems, which often rise in a non-expert markups:

1. *Complexity and ambiguity of the proposed tasks for non-expert crowdsourcing workers*
   For example, the task to markup commercial relevance for the commercial offer based on a client query could be a difficult and misleading task, as crowdsourcing workers can have very different opinion what a commercial relevance is, and even with good written instruction there is a chance of an ambiguous case, which was not covered in instruction. These statements also proved by these papers ([7], [8]), and even with experts, which are also have a bigger cost, dataset markup can face problems with noisy labels ([9]).

2. *Crowdsourcing cheaters and usage of bots, which automatically marks dataset in a random way*
   As crowdsourcing platforms works like a labor market, there is also a conflict between workers and employers, who can cheat the system and start markup datasets randomly or using some strategy: dataset markup can also suffer from a professional markup bots, which automatically markup datasets, evading anti-cheating rules or polices.

3. *Commercial shifts and changes on the crowdsourcing platforms*
   As a labor market, crowdsourcing workers choose the tasks, which they would do, from a bunch of projects, based on their compensation, time spending and any other significant parameters. As employers tries to get the most optimal labor force, they usually periodically change prices or any other significant parameters of the markup project, which can lead to a shift of labor force from or to a markup project, which can lead to a different distribution of workers on the project and different quality of the markup labels.

As a result of these problems, the ratio of corrupted labels in real-world datasets is reported to range from 8.0% to 38.5%, which is also proved by studies.[9], [10], [11].

Notation	Description
$\mathcal{X}$	the data feature space
$\mathcal{Y}$, $\tilde{\mathcal{Y}}$	the true and noisy space
$\mathcal{D}$, $\widetilde{\mathcal{D}}$	the clean and noisy training data
$P_{\mathcal{D}}$, $P_{\tilde{D}}$	the joint distributions of clean and noisy data
B_t	a set of mini-batch examples at time t
Θ_t	the parameter of a neural network at time t
$f(\cdot;\Theta_t)$	a deep neural network parameterized by Θ_t
ℓ	a specific loss function
$\mathcal{R}$	an empirical risk
$\mathbb{E}_{\mathcal{D}}$	an expectation over $\mathcal{D}$
x, x_i	a data example of $\mathcal{X}$
y, y_i	a true label of Y
$\tilde{y}$, $\tilde{y}_i$	a noisy label of $\tilde{Y}$
η	a specific learning rate
τ	a true noise rate
b	the number of mini batches in B_t
c	the number of classes
T, $\hat{T}$	The true and estimated noise matrix

Table 1. Summary of the notation

This paper reviews only methods for working with labeled data for a classification tasks. Classification is a supervised learning task for learning a function which maps an input features to a label. Let $\mathcal{X} \subset \mathbb{R}^d$ be the feature space and $Y = \{0,1\}^c$ be the ground-truth label space in a *one-hot* manner. In a standard classification problem, the training dataset is given $D = \{(x_i, y_i)\}_{i=1}^{N}$ which was obtained from an unknown joint distribution $P_{\mathcal{D}}$ over $\mathcal{X} \times \mathcal{Y}$, where each (x_i, y_i) is *independent and identically distributed*. The goal of classification is to learn mapping function $f(\cdot;\Theta)$: X → $[0,1]^c$ of the neural network parameterized by Θ such that the parameter Θ minimizes the empirical risk $\mathcal{R}_{\mathcal{D}}(f)$,

$$\mathcal{R}_{\mathcal{D}}(f) = \mathbb{E}_{\mathcal{D}}[\ell(f(x;\Theta),y)] = \frac{1}{|\mathcal{D}|}\sum_{(x,y)\in\mathcal{D}} \ell(f(x;\Theta),y)\,, (1.1)$$

where ℓ is a certain loss function.

But in a real-world scenarios data labels are corrupted, so neural networks are trained from noisy labels. This can be formulated as a noisy training dataset, obtained from a noisy joint distribution $P_{\tilde{D}}$ ˜ over $\mathcal{X}$ ×Y, where $\tilde{y}$ is a *noisy* label which may not be true. Hence, following the standard training procedure, a mini-batch $B_t = \{(x_i, \tilde{y}_i)\}_{i=1}^{b}$ comprising b examples is obtained randomly from the noisy training dataset $\widetilde{\mathcal{D}}$ at time t. Subsequently, the neural network parameter Θ_t at time t is updated along the descent direction of the empirical risk on mini-batch B_t,

$$\Theta_{t+1} = \Theta_t - \eta\nabla\left(\frac{1}{|\mathrm{B}_{t|}}\sum_{(x,\tilde{y})}(\ell f(x;\ \Theta_t),\tilde{y})\right), (1.2)$$

where η is a learning rate specified.

Here, the formula is no longer *noisetolerant* because of the loss calculated by the noisy labels. Neural networks can memorize corrupted labels easily and degenerate their generalizations on unseen data. Hence, mitigating the adverse effects of noisy labels is essential to enable noise-tolerant training for deep learning.

Scientific papers distinguish two types of label noise, which can affect markuped dataset [12]:

1. *Instance-independent Label Noise:*

   A usual approach for modeling label noise assumes that the mislabeled process is conditionally *independent* on data features. That is, true label is mislabeled by a *noise transition matrix* T $\in [0,1]^{c\times c}$, where

   $$T_{ij} := p(\tilde{y} = j|y = i), (1.3)$$

   is the probability of the true label i being flipped into a corrupted label j.

In this approach, the noise is called a *symmetric* (or *uniform*) noise with a noise rate $\tau \in [0,1]$ if

$$\forall_{i=j} T_{ij} = 1 - \tau \land \forall_{i \neq j} T_{ij} = \frac{\tau}{c-1}, (1.4)$$

where a true label is flipped into other labels with equal probability.

In contrast to symmetric noise, the noise is called an *asymmetric* (or *label-dependent*) noise if

$$\forall_{i=j} T_{ij} = 1 - \tau \land \exists_{i \neq j, i \neq k, j \neq k} \mathrm{T}_{ij} > \mathrm{T}_{ik}, (1.5)$$

where a true label is more likely to be corrupted into a particular label. For example, a “dog” is can be confused with a “cat” with bigger probability than with a “squarel.”

2. *Instance-dependent Label Noise:*

   For more realistic noise modeling, the mislabeled probability is assumed to be *dependent* on both the data features and class labels. Accordingly, the corruption probability is defined as

   $$P_{ij} = p(\tilde{y} = j | y = i, x), (1.6)$$

   It means, that the data feature of an example x affects the chance of x being mislabeled.

Most existing algorithms dealt with instance-independent noise, but instance-dependent noise has not yet been extensively investigated owing to its complex modeling. In this paper, we will assume that label noise is instance-independent. This actually can be misleading, as mislabeled errors usually occurs on a difficult or rare examples in the classes (often calls low density regions), which has higher probability of being mislabeled than for example, easy of medium difficulty tasks.

Chapter 2

Methods overview in literature

2.1 Classical machine learning methods

The problem of noisy labels has been an old challenge in classical machine learning [14], as this problem existed before neural networks became so popular. In classical machine learning label noise is closely related to outlier detection [15] and anomaly detection [16]. This comes with that fact that usually mislabelled instances are outliers in some kind of feature space, as their label has a low probability of occurrence. Similarly, such instances may also look anomalous, it’s common to treat the label noise problem as an anomaly detection problem and use anomaly detection techniques.

However, it’s not mandatory that mislabelled instances are outliers or anomalies. For example, if errors happen in a boundary region where all class labels have equal probability, the mislabelled instances are not rare events or can be spot by an anomaly method. Also, an outlier is not always a mislabelled sample, since it can be a low-probability event.

Machine learning methods can be categorized into three categories:

1. *Model selection or design methods*
   Conducted studies shows that different classical machine learning models and techniques are more or less tolerant to the label noise. The results of studies can be shown in the tables:

Model	**Robust to label noise**	**Reason**	**Study**
KNN	Less robust	1NN models tend to overfit to a misclassified instance, this effect can be negated by a correct choice of K	[14], [17]
Naïve Bayes	More robust	Conditional probability values are relatively less sensitive to data errors	[14], [17]
Logistic regression	Less robust	Log loss is vulnerable to a label noise (the same as hinge loss)	[14], [18]
Decision trees	Less robust	Single tree usually overfits data, can be eased by pre and post-pruning tree	[14], [17]
Random forest	More robust	Model architecture prevents from overfitting	[14], [17]
SVM	Less robust	SVM relies on each single instance to derive hyperplanes, so it could be easily altered by inclusion or exclusion of a single noisy instance	[14], [17]
Boosting algorithms	Less robust	On the later stages of learning, boosting algorithms overfits on misclassified labels	[14], [17]

Table 2: Label noise in machine learning models and techniques

Given this result, the choice of the more robust model can decrease the influence of label noise, but this comes with the choice of less complex and sensitive model, as the more robust models tends to less overfits the data.

2. *Filter-based methods*
   Another option is to find and filter mislabeled instances with some criteria. The results of following studies can be shown in the tables:

Method	**Description**	**Studies**
Threshfold filtering	Misclassified labels can be identified based on exceeding threshold on some conditional probability or complexity measure, which calculates if instance increases complexity of the model too much	[19], [20]
Classification filtering	Learn a classifier on a clean dataset (like SVM) and filter instances based on it’s predictions	[21], [22]
Voting filtering	Learn ensemble of classifiers and use cross-validation and some voting mechanism (like majority or consensus vote) to filter misclassified instances	[23], [24]
KNN-Based Methods (Graph-Based methods)	Use distances in feature space to identify misclassified labels, usually considering some number of neighbors in a local field	[25], [26]
Boosting-based methods	As boosting methods tends to give high weights to a mislabeled instances on the later stages of learning, algorithms try to remove percentage of the samples with the highest weights	[27], [28]
Other methods	Different methods try to remove elements which are close to classification boundaries or detect mislabeled instances by measuring the impact on the classification function using a leave-one-out strategy	[29], [30]

Table 3: Filter-based machine learning methods

One of the advantages of these methods is that removed instances have absolutely no effects on learning process. Removed labels can be sent to resubmit to a human expert for relabelling. However, this can be too costly and there is no guarantee that the new labels will actually be noise-free.

Also these methods have common drawback, that they could filter correctly labeled instances, which decreases training dataset. This overcleansing problem is particularly important for imbalanced datasets, as minority instances have bigger probability to be removed by classification filtering. Label noise cleansing can also reduce the complexity of inferred models, as filter methods could remove more complex instances from the dataset.

3. *Methods that aim to model label noise in a parallel with training the model*
   Methods in this category learns to determine and downweight mislabeled instances in parallel with classifier training model training. The results of these studies can be shown in the tables:

Method	Description	Study
Probabilistic Methods	Methods tries to include prior knowledge, using Bayesian approach, frequentist approaches and clustering-based methods to effectively correct mislabeled instances during training	[31], [32], [33]
SVM with robust losses	Usual SVM algorithm is modified, for example, k samples are allowed to be not taken into account in the objective function or adding another set of trainable weights to the loss to down-weight mislabeled instances	[34], [35]
Boosting algorithms with modifications	Methods change initial boosting algorithm, which gives large weights for mislabelled instances in late stages of learning, to some sort of smothing algorithm, which combines bagging and boosting on the training data, to decrease the sensitivity of the weights update function	[36], [37]

Table 4: Machine learning methods that aim to model label noise in a parallel with training the model

The advantages of probabilistic models are that they can take advantage of prior knowledge of the problem and they use whole training dataset. However, the main problem of this approach is that they increase the complexity of learning algorithms which leads to overfitting, because of additional parameters in the model. As complete model consists of a label noise model and a classification model, there is a problem that only one part is useful for prediction.

2.2 Deep learning methods

Deep learning models are usually trained on a significantly bigger datasets than classical machine learning models. Consequently, these datasets are usually labeled by non-experts on a crowdsourcing platform rather than by experts. That’s why, the label noise level in these datasets is usually much higher compared with the datasets from classical machine learning.

Recent studies confirmed that label noise can affect the quality of deep neural network model in different ways:

1. *If the task is not difficult (like image classification) and learning of the neural network is stopped without overfitting, neural networks can be tolerant to a label noise*

According to the studies [38] and [39] deep learning models tends to learn patterns in the data at least at the start of training, instead of memorizing data instances, also usual explicit reguralization techniques, like dropout, helps preventing memorization of the data. This works on a classical deep learning tasks, like MNIST or CIFAR100 classification, but for a more complex task, like face recognition, study [40] shows that label noise had a noticeable impact on the quality of the model, as training on a small, clean dataset gave better results than using large dataset with significant label noise.

2. *Label noise influence depends on the structure of the data and it’s diversity, model architecture, training procedures and the amount of label noise.*

According to the study [41] deep learning models are usually much more vulnerable to label noise that is clustered (in some label class or in a feature space), rather than the label noise which spreads uniformly across all classes .

Different strategies, like changing model architecture and training procedures will be reviewed latter in the chapter.

3. *Real-world models usually show severe degradation because of noisy labels*

According to the studies [42], [43], [44] real-world applications, like medical image analysis, transfer learning, facial attribute recognition, suffers significantly from label noise: the reason behind this is that these applications have much more complex structure, that can easily overfit on a mislabeled instances. Also, datasets for real-world applications are not so clean as usual, “academic” datasets, which has significantly less percentage of mislabeled instances than real-world datasets.

Therefore, the topic of noisy labels and methods of dealing with was studied in many study papers in the last years. In this paper these methods organized under six categories. The methods can overlap to more than one category.

Category	**Methods**	**Study**
Label cleaning and pre-processing	a) Training filtering model on a small clean dataset b) Feature vector comparison and filtering, using CleanNet c) Classification confidence filtering d) GAN modeling to filter label noise	a) [45], [46] b) [47], [48] c) [49]. [50] d) [51], [52]
Network architecture	a) Adding special noise layer b) Adding annotators matrices c) Adding probabilistic models	a) [53], [54] b) [55] c) [56], [57]
Loss functions	a) Box-cox modification of MAE and iMAE b) Adding possibility of abstention some data points to a loss function c) Adding confusion matrix to correct wrong labels or predictions	a) [59], [60] b) [61], [62] c) [63]
Data reweighting	a) Assign weights to training instances by minimizing loss on a small, clean dataset b) Removing a fraction of data from training epoch, where loss is the largest c) Learning reweighting scheme from training data	a) [64], [65] b) [66] c) [67]
Data and label consistency	a) Include information of correlation between features learned by the layers of the model b) Compare noisy instances with the	a) [69] b) [70] c) [71]

	representation of classes c) Mini-Batch training with the same class label instances using BundleNet	
Training procedures	a) Learning based on increased difficulty (Curriculum learning) b) Knowledge distillation approach c) Co-teaching d) Training only on instances, where predicted and noisy labels are the same e) Changing learning rate, batch size and other settings	a) [72], [73] b) [74], [75] c) [76], [77], [78] d) [79] e) [80]

Table 5: Main methods of working with noisy labels in deep learning

1. **Label cleaning and pre-processing**

   Methods in this category tries to fix or discard training instances that are identified as mislabeled labels. This process can be done before training or in parallel with the main training. Usually these methods is used, when you have small clean dataset.

   a) Training filtering model on a clean dataset

   If you have small clean dataset, the idea which is used in many methods is to train on a big noisy dataset and then fine-tune on a clean dataset. Paper [45] proposed another method: to train another CNN model which will share the feature extraction layers with the main model. This CNN trains on the clean labels to identify noisy labels. Updated dataset is used in the training of the main model. Paper [46] develops this approach, by training an ensemble of classifiers (with cross-validation) and then using the results of the ensemble as soft labels to learn the final classifier.

   b) Feature vector comparison and filtering, using CleanNet

   CleanNet, proposed by [47], develops previous approach by replacing classification of possible noisy labels to a comparison of feature vector from a training instance with a feature vector, which represents the given class. The class vectors can be obtained by aggregating features from the clean dataset.

Usual cosine similarity between noisy and clean vectors is used to determine if the instance was mislabeled or used to determine weights of the instance. The develop of this approach was introduced in the paper [48], where several representative vectors can be used for every class.

c) Classification confidence filtering

Classical confidence filtering model is a Rank Pruning algorithm, proposed by [49], which identifies training instances with confident labels and uses only those data points to update the classifier. This method uses the hypothesis that training instance which has predicted probability closer to 1 has bigger probability to have correct labels. This method has another version when data instances are removed if they are least confident. Paper [50] develops Rank Pruning algorithm by adding iterative filtering approach. This method tries to estimate prediction uncertainty by using methods like Deep Ensembles and Monte-Carlo dropout in parallel with training and corrects data instances that are predicted to be mislabeled.

d) GAN modeling to remove label noise

Another approach to clean noisy labels is to use GANs. In paper [51] researchers train two generators models to generate examples which are considered by the discriminator as the most positive, respectively most negative, with the highest confidence as possible and then uses these confidences to relabel initial noisy labels. This approach gives better results than Rank pruning algorithm on a smaller datasets.

2. **Network architecture**

Methods in this category changes usual deep neural network architecture to make it more tolerant to a noisy labels.

a) Adding special noise layer

This method proposes to add a “noise layer” as a last layer of the main network. This noise layer can be a transition matrix between noisy and true labels, which idea was reviewed by the paper [53]. The transition matrix is learnt in parallel with main network weights on back-propagation. A GAN version of this layer was proposed by [54].

b) Adding annotators matrices

Paper [55] develops previous approach by adding annotator confusion matrices which are learnt in parallel with the main model. In this approach model results multiplied by corresponding annotators matrices and then model loss includes adjusted by annotators class probabilities and probabilities of respective annotators. All main model and annotator matrices weights can be learned using SGD. This method can model different types of annotators confusion matrices for different annotators types.

c) Adding probabilistic graphical models

Another approach is to integrate probabilistic graphical models into the main model to model label noise. Paper [56] proposed a probabilistic directed model which extends initial CNN and helps to predict the true labels and uses them to supervise the training of the network. The method uses a small clean dataset. Paper [57] develops this idea by adding undirected graphical model which learns the relationship between correct and mislabeled instances, which can be learnt without clean dataset and it also allows to incorporate domains specific sources of information.

3. **Loss functions**

Methods in this category doesn’t change training data, training procedures and model architecture and focuses only on loss function.

a) Box-cox modification of MAE and iMAE

Traditional loss functions, like MAE, MSE and cross-entropy, are studied in the paper [58]. The paper shows that MAE is robust to label noise, while cross-entropy and MSE are not. This is because cross-entropy gives bigger weights on hard examples (which is defining property for a clean datasets), while MAE evaluates all data instances more equally.

For a multi-class classification problem, denoting the vector of true and predicted probabilities with $p(y = j|x)$ and $\hat{p}(y = j|x)$, respectively, the loss function of cross-entropy is defined as:

$$L_{ce} = \sum_{j} p(y = j|x) log \hat{p}(y = j|x) \quad (2.1)$$

The MAE loss is defined as:

$$L_{MAE} = \sum_{j} |p(y = j|x) - \hat{p}(y = j|x)| \quad (2.2)$$

However, paper [59] shows that MAE suffers from longer times for training and smaller test accuracy, so modified loss function was proposed based on Box-Cox transformation:

$$L_q(f(x), e_j) = \frac{(1 - f_j(x)^q)}{q} \quad (2.3)$$

where q ∈ (0, 1].

Another paper [60] examines the gradients of cross-entropy and MAE loss functions, which advantages and disadvantages they have. Based on these conclusions they proposed an improved MAE (iMAE) that changes MAE's sample weighting strategy. Specifically, they showed that the L1 norm of the gradient of LMAE with respect to the logit vector leads to down-weighting of difficult but informative data samples. To fix this, they suggested to transform the MAE weights nonlinearly with a new weighting defined as:

$$\exp\left(Tp(y|x)\right)\left(1 - p(y|x)\right) \quad (2.4)$$

where the hyperparameter T was set equal to 8 for training on a dataset with noisy labels.

b) Adding possibility of abstention some data points to a loss function

Paper [61] proposed another approach of modifying standard loss functions by adding abstention penalty in the cross-entropy. The proposed modification allows the model to omit predictions on some data instances at the price of abstention penalty. Another idea was proposed in [62] as a trimmed cross-entropy loss, which simply ignores the training samples which has the biggest loss values, as well-trained model usually will give high loss values to a mislabeled data instances.

c) Adding confusion matrix to correct wrong labels or predictions

Paper [63] proposed another two ways which improves the ability to negate label noise for a loss function when training deep learning models. These methods uses the error confusion matrix T, which is defined as

$$T_{i,j} = p\left(\tilde{y} = e^{j} \middle| y = e^{i}\right) \ (2.5)$$

where $\tilde{y}$ are noisy labels and y are true labels.

Assuming T is non-singular, the first correction strategies, which was called «backward correction» is:

$$l_{corr}(\hat{p}(y|x)) = T^{-1} l\big(\hat{p}(y|x)\big) (2.6)$$

This correction is a linear weighting of the loss values for each possible label, where the weights, given by T, are the probability of the true label given the observed label.

The alternative approach, named forward correction, is based on correcting the model predictions. The corrected loss is defined as

$$l_{corr}\big(h(x)\big) = l(T^{T} \psi^{-1} \Big(\big(h(x)\big)\Big) \ (2.7)$$

where h is the vector of logits, and ψ^{-1} is the inverse of the link function for the loss function in consideration, which is the standard softmax for cross-entropy loss. The authors show that both these corrections make loss functions unbiased:

$$\forall x E_{\tilde{y}|x} l_{corr} = E_{y|x} l \ \ (2.8)$$

4. **Data re-weighting**

Methods in this category tries to down-weighting training instances which has probability to be mislabeled.

a) Assign weights to training instances by minimizing loss on clean dataset

Paper [64] proposed metalearning approach, which weights the training data. This method trains the weights on a clean dataset and then assign weights to the main training dataset.

The develop of this approach was proposed in [65], where samples re-weight strategy is changing based on optimization gradient re-scaling of the main dataset.

b) Removing a fraction of data from training epoch, where loss is the largest

Paper [66] proposed a training strategy that can be interpreted as a form of data re-weighting. The idea is to remove a sample of training data with the biggest lost and update the model only on the remaining training instances. This method assumes common idea, that mislabeled training instances usually gets high loss values as the train progresses.

c) Learning reweighting scheme from training data

Paper [67] propose to use a simple fully-connected model with a one hidden layer to learn weighting strategy instead of assuming a pre-defined weighting scheme. The MLP in this method is pretrained on a small clean dataset. The experiments with this method also shows that this kind of model down-weight instances with large loss function from the main model.

5. **Data and label consistency**

Methods in this category works with the hypothesis that most of the training data instances have correct labels and there is correlation between same class instances (or the features from the layers of the network). These correlations can be used to reduce label noise.

a) Include information of correlation between features learned by layers of the network

Paper [69] shows the method which uses the correlation of the neural network features. The idea is that network features from layers,which were learned by model on data instances of the same class should have high correlation. All instances with the same class then fed into a light neural network which assigns confidence weights to each training instance based on the probability that it has correct label. These weights then are used to compute a representative feature vector for classes, which then can be used to train the main classification model.

b) Compare noisy instances with the representation of classes

Paper [70] propose a method with name “auxiliary image regularization”. This method requires a small clean dataset. The idea is to boost consistency between training images and clean images. Group sparsity metric were added as a term to loss function which boosts the features of a training instances to be closer to a small number of clean images.

c) Mini-butch training with the same class label instances using BundleNet

Paper [71] proposed BundleNet, where training instances with the same labels were grouped and fed to the network as a single input. The method looks like standard mini-batch training, but it empirically improves accuracy on image classification with noisy labels. Model uses similarity between instances to identify and down-weight instance with higher probability of being mislabeled.

6. **Training procedures**

Methods in this category are very diverse. Some of them are based on machine learning methods, while others focus on modifying the settings of the training pipeline such as learning rate and regularization.

a) Learning based on increased difficulty (Curriculum learning)

Paper [72] proposed curriculum learning which trains a model based on increasing complexity or difficulty. There are two network in this model, the first one, called Mentor-Net, gives a curriculum, which is a weights on the training samples, and a second one, called Student-Net, which uses these weights while learning.

The develop of this approach was proposed in [73] where model tries to cluster the training instances in feature space and identifies instances that have bigger probability to be mislabeled as they come to low-density clusters.

b) Knowledge distillation approach

Another approach was introduced in [74], called knowledge distillation approach. These methods use pseudo-labeling, as a combination of noisy label and label which is predicted by auxiliary model, which was trained on a clean dataset.

Paper [75] also proposed a combination of predicted labels and noisy but on its current training stage. As the model learns, it becomes more accurate, and its predictions can be weighted more strongly while down weighting original labels.

c) Co-teaching

A number of studies have proposed methods involves training more than one model. Paper [76] suggested training two networks with the same architecture but with random initialization, and updates the network parameters only when predictions differs.

This method evolves into co-teaching [77], where the two networks identify noise-free instances in their batches and gives this updated information to the other network. Co-teaching was further developed by [78], where model focuses on data samples with lower loss values as it helps to reduce the risk of training on mislabeled instances.

d) Training only on instances, where predicted and noisy labels are the same

Paper [62] showed another method, which is more effective than co-teaching if label noise is significant: in the proposed method, the training instances were divided into two datasets. The model iteratively trains on one dataset and then is tested on the other. If predicted and noisy labels agreed, training instance is assumed to have the correct label and it’s added to the training dataset.

e) Changing learning rate, batch size and other settings

Another approach is to simply modify the learning rate, batch size, or other settings in the training methodology.

Paper [157] proposes training strategies, like ordering clean and noisy dataset, proper learning rate adjustments and batch size adjustments. For example, with strong label noise, the effective batch size should decrease with a proper scaling of the learning rate.

Paper [76] proposed to mixup instances from a noisy dataset and a clean dataset in each training epoch, giving bigger weights to the instances with clean labels.

Chapter 3

Datasets overview

As mentioned in the introduction, the analysis of the selected methods will be based on real datasets, which were collected in Yandex Toloka. Yandex Toloka is an open crowdsourcing platform for dataset markup, which can be used by anyone, both as a worker, who markup tasks, and as employer, who gives tasks. The platform also provides tools for markup quality control such as automated rules and tasks-based quality control.

Further in the text, the following concepts will be used:

• **Project** - a set of datasets markups in Yandex Toloka, united by one topic and instructions.

• **Tolokers** – markup workers who can freely come, select and complete tasks and leave the crowdsourcing platform and are not on the staff of Yandex.

• **Assessors** - more experienced markers who are on the Yandex staff and who perform more complex markings, adjust markup projects, and prepare honeypots.

• **Task** - a separate task (one instance from the dataset) in the Toloka interface, which is performed by the tolokers

• **Honeypots** - test tasks, which do not differ for tolokers in any way from ordinary tasks, but according to which the quality of marking markers is further considered

The main tool for monitoring the quality of markup are honeypots: Based on the ratio of correctly marked honeypots to the total amount of marked honeypots (% of correct answers), the toloker is given the markup skill for a specific project, which determines his earnings for a separate task and the ability to mark tasks further in this project.

The honeypots are prepared by the assessors: it is assumed that their quality is close to the ground truth, therefore, in the future, the honeypots prepared by the assessors will be used as the ground truth of the selected datasets.

Next, it is necessary to describe the process of collecting and marking honeypots, from which it follows the possibility of using datasets based on honeypots, as a basis for analyzing noisy datasets:

1. Tolokers completing tasks that are given to them by customers
2. From what the tolokers have completed, a sample of tasks (for example, 500 tasks) is taken at a certain frequency (about a week or a month) and they are completed (marked) by assessors
3. Further, the marked tasks are added to the general pool of honeypots and in further pools on these honeypots it already assesses the quality of the marking of the tolokers

Thus, datasets consisting of honeypots have both noisy labels (responses from tolokers) and ground truth labels (responses from assessors), which allows using such datasets in analyzing the effectiveness of methods for working with noisy labels, without using artificial noise generation methods, such as a random change of the label or the main one on any distribution or model [13].

Next, the two datasets that have been selected will be described.

3.1 Pictures dataset

Picture dataset was collected by a picture markup in Yandex Toloka, where the task was to determine the visual quality of the picture from the task.

Tolokers has 3 answer options:

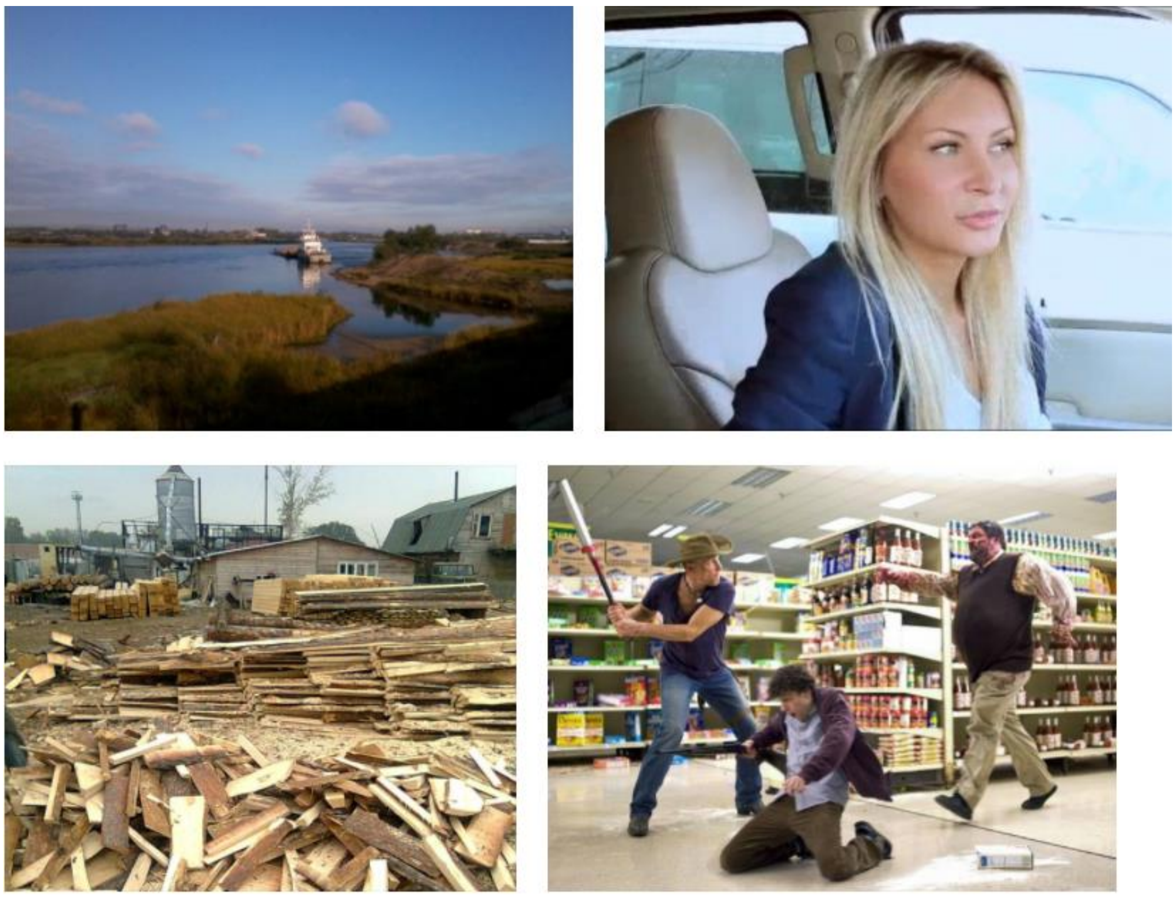
1. **Excellent quality -** The picture of "excellent" quality is distinguished by the fact that it looks quite professional, has good sharpness and balance of brightness, important details are clearly distinguishable. These pictures are most often pleasant or interesting to view or contain some kind of visual information in an easy-to-read form.

   Excellent quality pictures can be of different types, here are some typical examples:

Pic 1: examples of excellent quality pictures

2. **Medium quality** - is a picture without obvious defects and with well-discernible content. Most of everyday amateur photographs and children's drawings that are of little interest to outsiders and of little use for use (decorative, educational, etc.) fall into this category, as well as professional photos, illustrations, screenshots, the quality of which has been noticeably reduced because of excessive compression file, stretching the size of the picture or adding a watermark that does not interfere too much.

   Here are some typical examples of a medium quality:



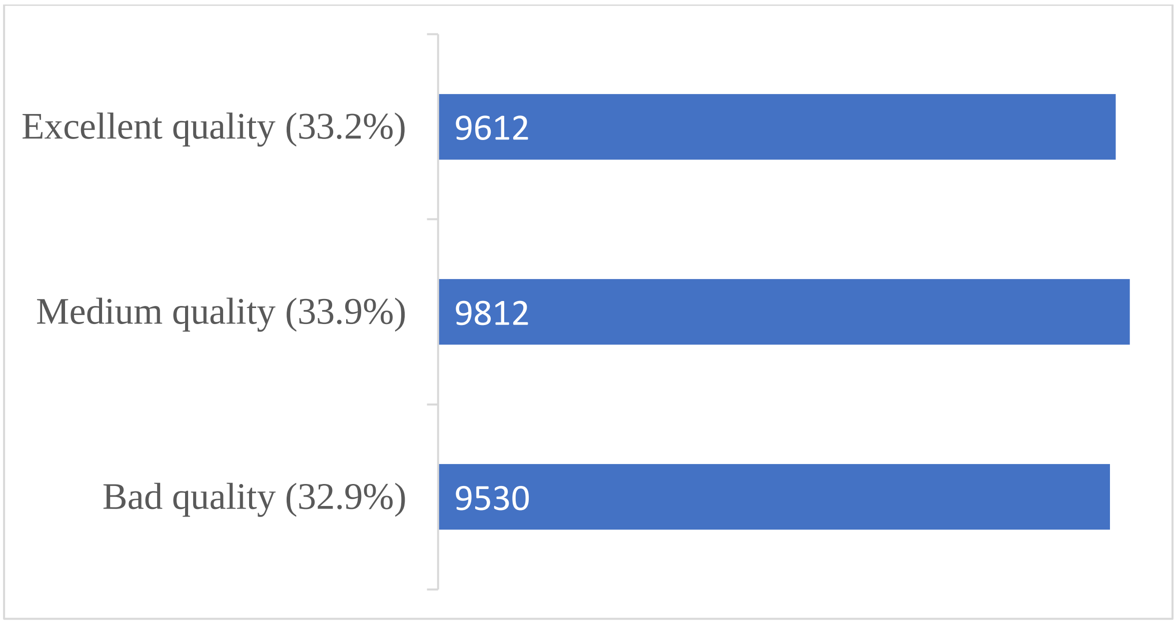
Pic 2: the examples of medium quality pictures

3. **Bad quality -** if the picture is heavily blurry (at the size shown), noisy, darkened, blown out, intentionally spoiled (except when it is obviously an artistic trick), contains a large and noticeable watermark that gets in the way, containing gross compression artifacts. It also includes pictures that are too small for their purpose: unreadable documents, diagrams, maps, tables

   Here are some typical examples of a bad quality:

Pic 3: examples of a bad quality pictures

There are 28 954 honeypots in the dataset. The distribution of labels in the dataset can be viewed in the following graph:



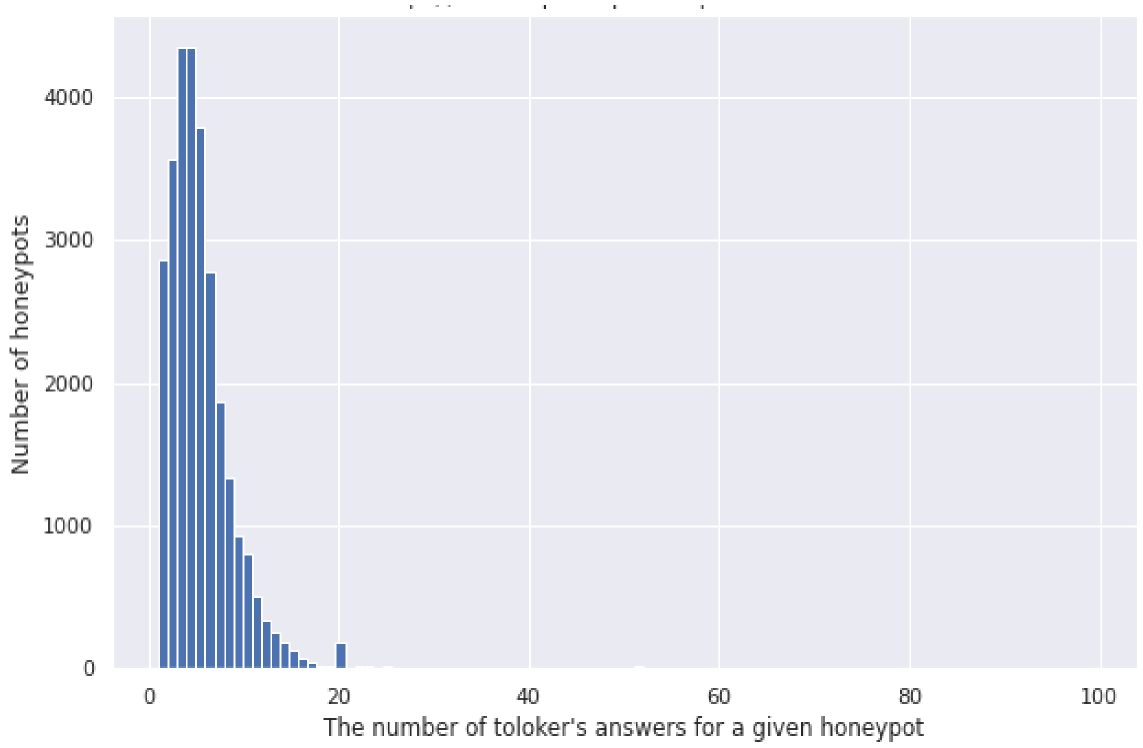
Pic 4: Distribution of honeypots in the picture dataset

For these honeypots, there are 242 159 tolokers answers. The main characteristics for this dataset are presented in the table:

Characteristic	**value**
Number of honeypots	28 954
Number of toloker's answers	242 159
Mean number toloker's answers for a given honeypot	8.36
STD number toloker's answers for a given honeypot	31.8
Min number toloker's answers for a given honeypot	1
25 percentile toloker's answers for a given honeypot	3
50 percentile toloker's answers for a given honeypot	4
75 percentile toloker's answers for a given honeypot	7
Max number toloker's answers for a given honeypot	1 044

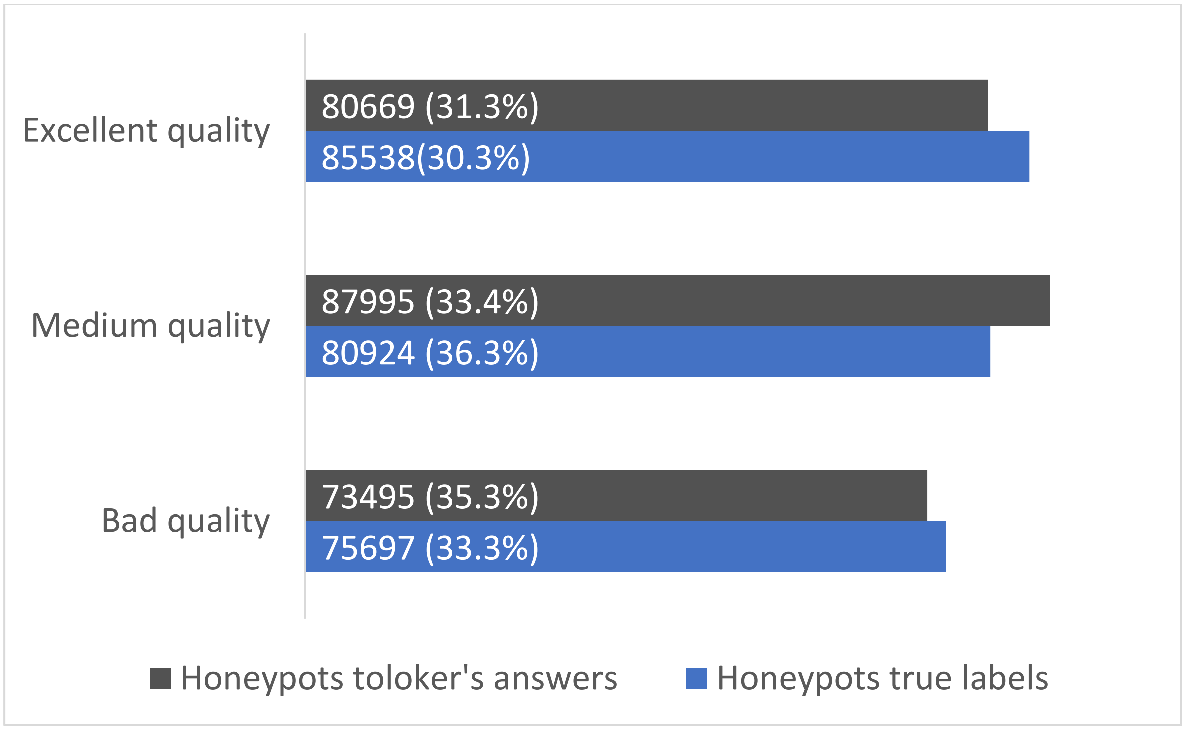
Table 6: Main characteristics of the picture dataset

The distribution of toloker's answers quantity for honeypots in the pictures dataset can be shown on graph:



Pic 5: The distribution of toloker's answers quantity for the pictures honeypots

To show, that there is a label noise in the dataset, we can look at the the difference between true honeypots and toloker's answers.



Pic 6: Distribution of used honeypots (honeypots can be used several times) and toloker's answers on these honeypots in the picture dataset

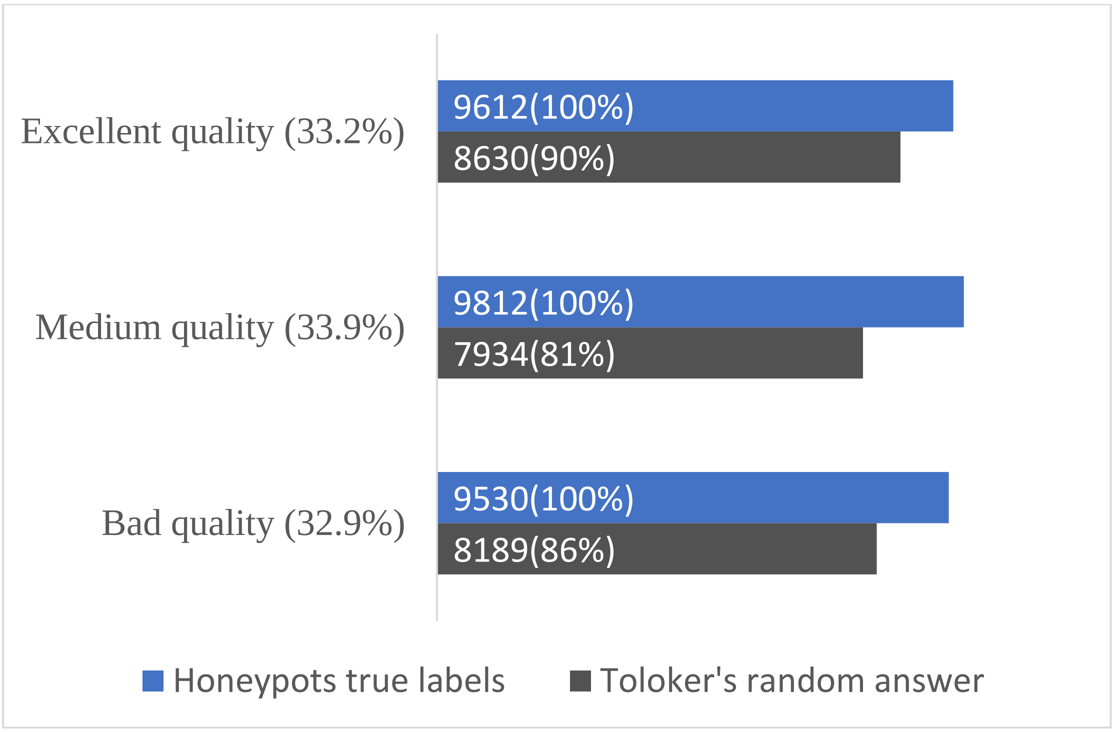
In the ideal scenario the honeypots true labels distribution should match with the honeypot's toloker's answers, but the graph shows, that there is some sort of noisiness in the dataset.

To increase the markup quality, projects usually use an overlap and some sort of aggregation on tasks (the task is sent to a several tolokers and then the raw answers aggregated based on some rule, for example, simple majority vote).

To measure the noisiness of the picture dataset I compared assessors true labels and toloker's answers with two different approaches:

- *compared the answer of the assessor (ground_truth) and the random answer of the toloker from history*:

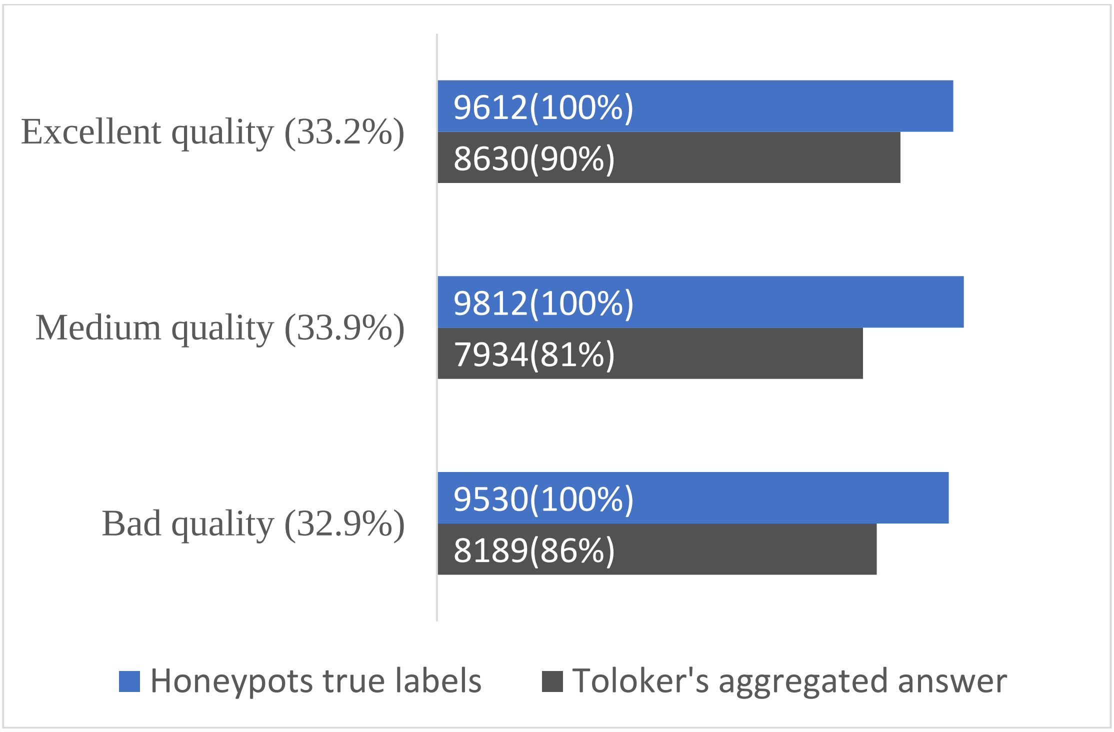
  It gives 77.6% matches. The distribution of the corrected answers can be showed on the graph:



Pic 7: Distribution of honeypots in the picture dataset and toloker's answers on these honeypots, using random answer from the answers history

- *compare the answer of the assessor (ground_truth) and the aggregated answer of the toloker (chose the most frequent from history)*

  It gives 85.4% matches



Pic 8: Distribution of honeypots in the picture dataset and toloker's answers on these honeypots, using aggregated answer based on the answers history

Since the aggregating algorithm is used in real projects, for further analysis methods for working with noisy labels, the aggregated answers of tolokers will be used. Thus, the estimated noise in the dataset is 14.6%

3.2 Text dataset

Text dataset was collected by a text markup in Yandex Toloka, where the task was to determine if the request is visually commercial or not. Visual commerce was defined as a situation in which the user usually first searches for pictures of goods in the Internet, if necessary, also looks at alternative options, and then makes a decision to buy the goods.

Tolokers has 2 answer options in this markup:

- *visual commercial*

This label can contain different categories of goods which can be seen in the table:

Category	**Examples**
Clothing	Dresses, shoes,
Electronics	Mobile phones, computers, laptops, audio and video equipment,
Sporting goods	Skis, bicycles, rollers, protection for any kind of sport.
Goods for kids	Strollers, baby toys
Home interior items	Wallpaper, tile

Table 7: Examples of visual commercial categories for a text dataset

This list is not exhaustive. At the same time, even in these categories there may be non-visually commercial requests, if its visual component is not important for these products, for example, various parts for electronics or a machine.

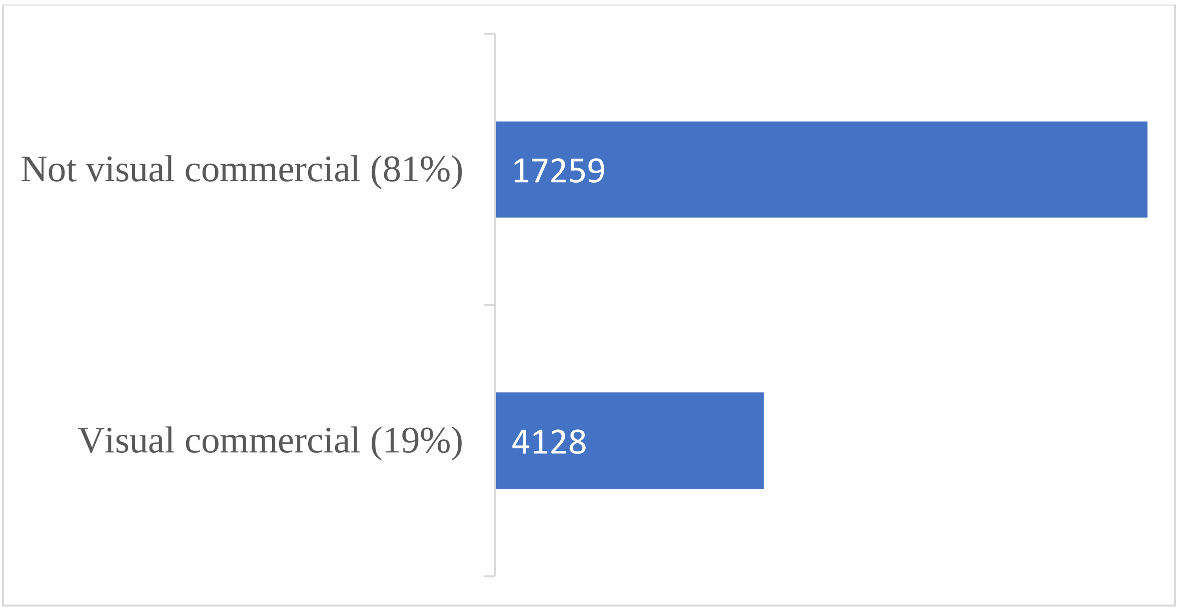
- *not visual commercial*

This label can contain different categories of non-visual goods and non-commercial queries which can be seen in the table. This list is not exhaustive and can be also expanded for noncommercial queries.

Category	**Examples**
Cosmetics and perfumery	Perfume, shampoo
Construction and repair	Glue, polyurethane foam
Spare parts	Spare parts for electronics, any technique and inventory
Goods for kids	Baby food

Table 8: Examples of visual commercial categories for a text dataset

There are 21 387 honeypots in the dataset. The distribution of labels in the dataset can be viewed in the following graph:



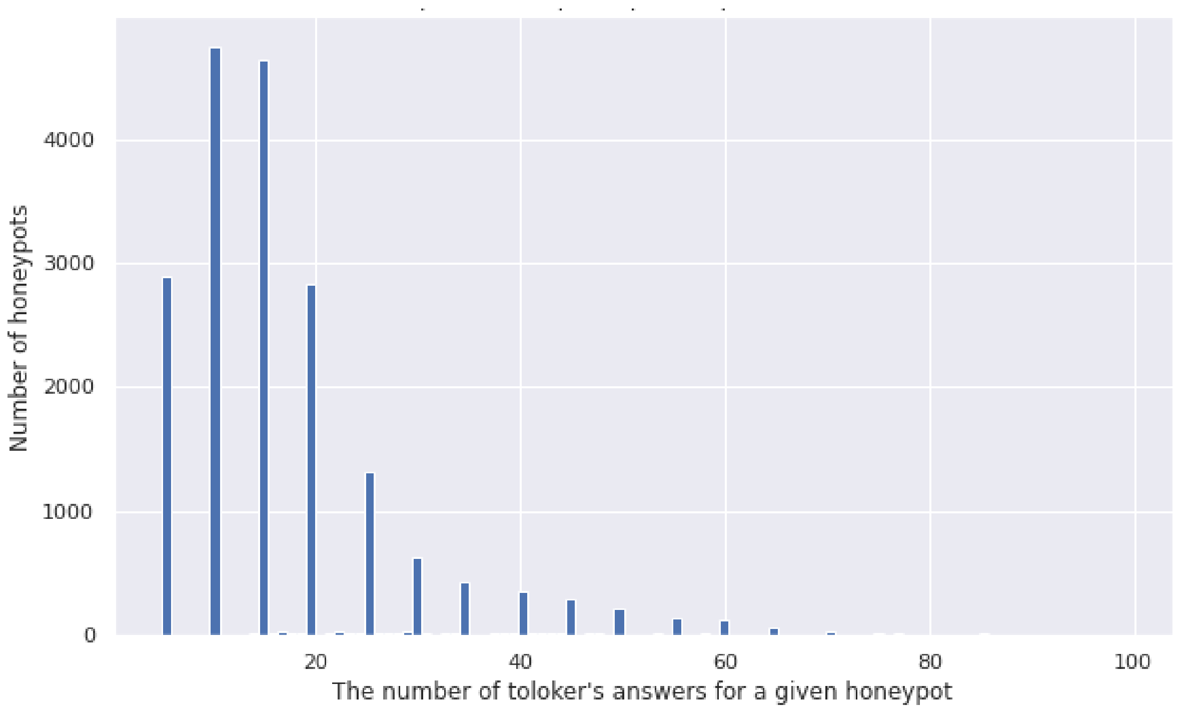
Pic 9: Distribution of honeypots in the text dataset

For these honeypots, there are 1 446 328 tolokers answers. The main characteristics for this dataset are presented in the table:

Characteristic	**value**
Number of honeypots	21 387
Number of toloker’s answers	1 446 328
Mean number toloker’s answers for a given honeypot	67.6
STD number toloker’s answers for a given honeypot	818.2
Min number toloker’s answers for a given honeypot	5
25 percentile toloker’s answers for a given honeypot	10
50 percentile toloker’s answers for a given honeypot	15
75 percentile toloker’s answers for a given honeypot	25
Max number toloker’s answers for a given honeypot	35 798

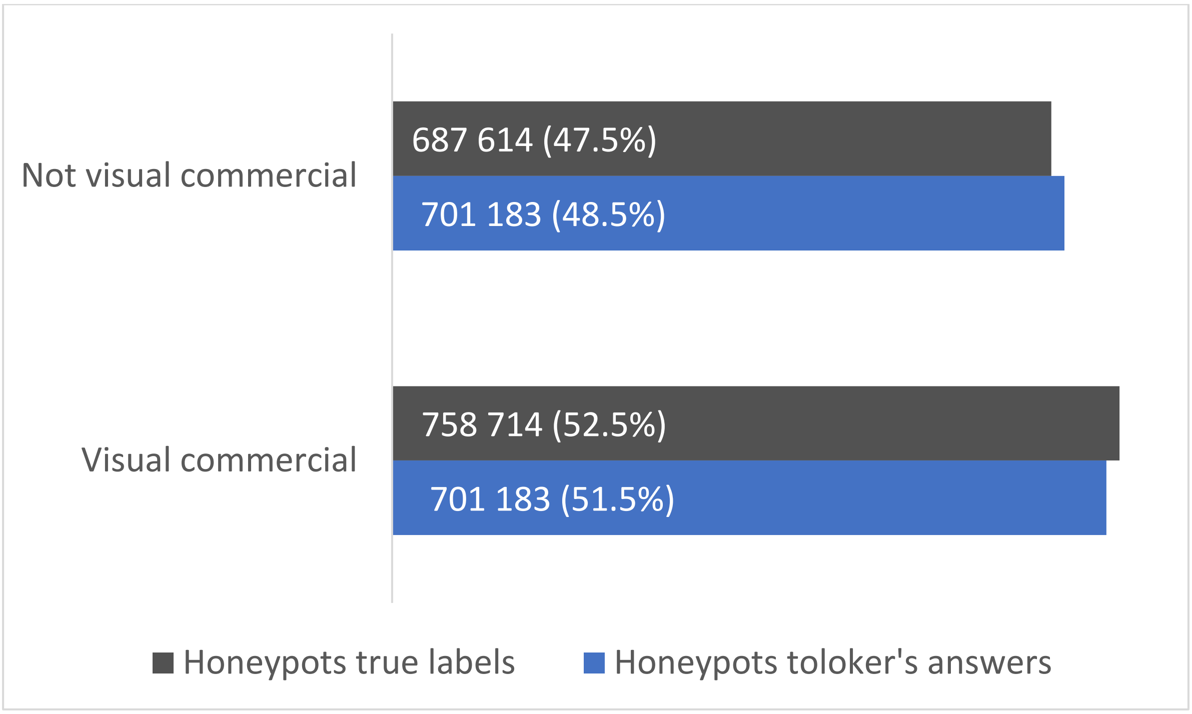
Table 9: Main characteristics of the text dataset

The distribution of toloker’s answers quantity for honeypots in the pictures dataset can be shown on graph:



Pic 10: The distribution of toloker’s answers quantity for the text honeypots

To show, that there is a label noise in the dataset, we can look at the the difference between true honeypots and toloker’s answers.

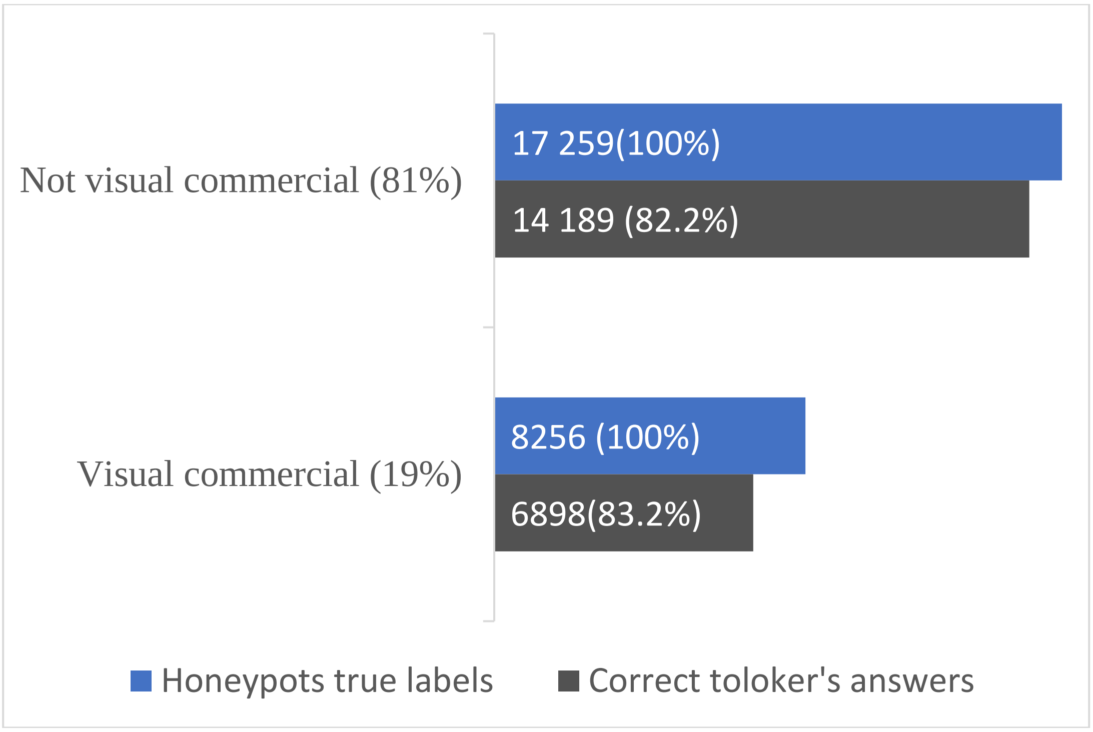


Pic 11: Distribution of used honeypots (honeypots can be used several times) and toloker’s answers on there honeypots in the text dataset

In the ideal scenario the honeypots true labels distribution should match with the honeypot’s toloker’s answers, but the graph shows, that there is some sort of noisiness in the dataset.

By analogy with a picture dataset, to measure the noisiness of the picture dataset I compared assessors true labels and toloker’s answers with two different approaches:

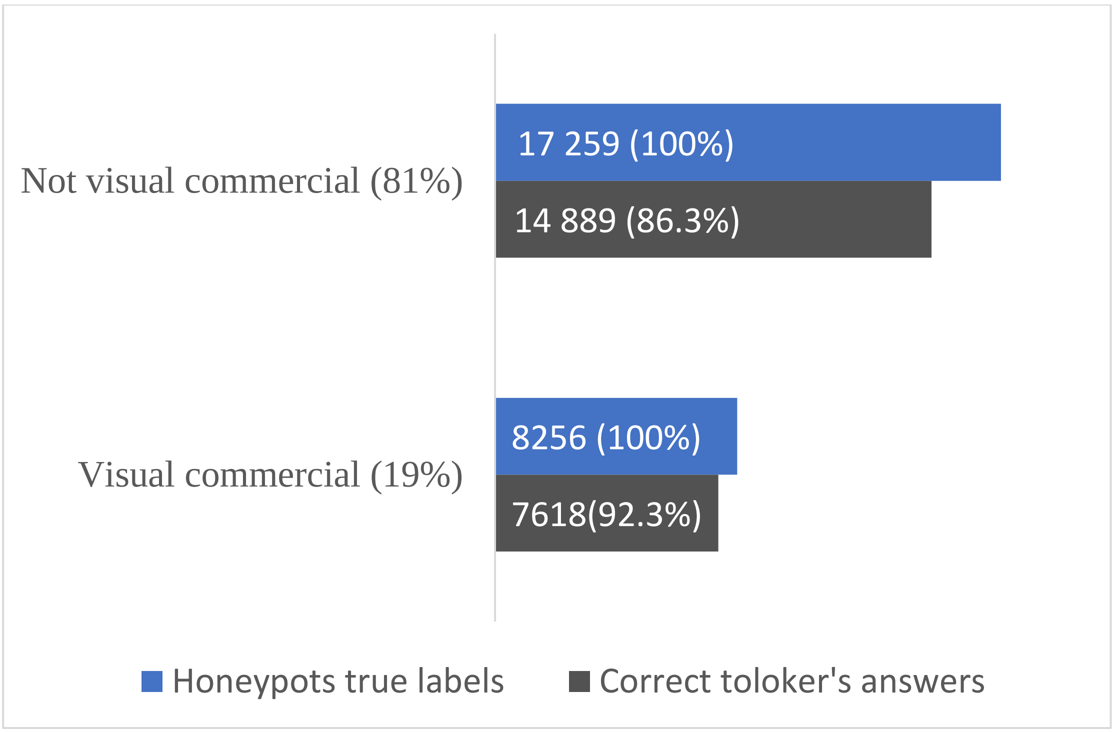
- *compared the answer of the assessor (ground_truth) and the random answer of the toloker from history*:
  It gives 82.3% matches. The distribution of the corrected answers can be showed on the graph:



Pic 12: Distribution of honeypots in the text dataset and toloker’s answers on these honeypots, using random answer from the answers history

- *compare the answer of the assessor (ground_truth) and the aggregated answer of the toloker (chose the most frequent from history)*

  It gives 87.3% matches



Pic 13: Distribution of honeypots in the text dataset and toloker’s answers on these honeypots, using aggregated answer based on the answers history

Since the aggregating algorithm is used in real projects, for further analysis methods for working with noisy labels, the aggregated answers of tolokers will be used. Thus, the estimated noise in the dataset is 12.7%

Chapter 4

Comparison of the chosen methods on datasets

4.1 Benchmark without using any methods

Pictures dataset

The purpose of the picture markup was to determine the visual quality of the image from the task. This problem can be formulated as a classification task, with three answers (“Bad quality”, “Normal quality”, “Excellent quality”).

To model this dataset, transfer learning with EfficientNet backbone was used, with several linear layers at the end.

EfficientNet were introduced in 2019 and quickly became state of the art model for computer vision, giving better results than ResNet or MobileNet.

The idea of EfficientNet is to scale all dimensions of image depth, width and resolution using a compound coefficient. Unlike conventional practice that arbitrary scales these factors, the EfficientNet scaling method uniformly scales network width, depth, and resolution with a set of fixed scaling coefficients.

The compound scaling method is justified by the intuition that if the input image is bigger, then the network needs more layers to increase the receptive field and more channels to capture more fine-grained patterns on the bigger image.

EfficientNet has several architectures, which has different input image size and the complexity of the model (number of modules in every block of the model, different number of modules and etc.).

To determine which versions of EfficientNet should be used, several neural networks were trained on a ground truth labels (assessors answer). Then, the model with the biggest accuracy score were selected, as the class distribution in the picture dataset is balanced: 33.2% percent of dataset instances are excellent quality, and 33.9% are normal quality and 32.9% are bad quality, so accuracy score can be used as a benchmark metric, as it treats all classes equally.

EfficientNet version	Neurons of last FC layer	Accuracy
b0	128	0.685
b1	128	0.681
b2	128	0.669
b3	128	0.687
b0	256	0.678
b1	256	0.680
b2	256	0.672
b3	256	0.666

Table 10: comparison of the models with different hyperparameters for a benchmark for picture dataset

The highest accuracy was got by efficientNet-b3 with 128 neurons on the last layer, with accuracy score of 0.687. This model will be used as an upper-bound, as this or greater results model should be obtained, if labels were clean. The classification report for this model on a ground-truth labels can be shown on the next table (threshold were selected by maximizing geometric mean of sensivity and specificity).

Label	Precision	Recall	f1-score	Num of instances
Good quality	0.757	0.660	0.705	509
Ok quality	0.651	0.559	0.602	484
Bad quality	0.757	0.660	0.705	439

Table 11: classification report for a ground-truth model for a picture dataset

Next, the model with the best hyperparameters were trained and validated on noisy labels (tolokers answers) and tested on ground truth labels. This model will be used as the lower-bound, as this model wasn't designed to handle label noise. This model got accuracy score of 0.643. The classification report for this model on a ground-truth labels can be shown on the next table.

Label	Precision	Recall	f1-score	Num of instances
Good quality	0.695	0.705	0.702	509
Ok quality	0.616	0.562	0.588	484
Bad quality	0.612	0.663	0.636	439

Table 12: classification report for a model without any label noise correction for a picture dataset

Hence, in this experiment the upper-bound for accuracy score for the text dataset is 0.687 and the lower-bound is 0.643.

Text dataset

The purpose of the text markup was to determine if the user query is visually commercial or not. This problem can be formulated as a binary classification task.

As users queries usually don't have big length, bidirectional LSTM model with pretrained embeddings were selected as a benchmark model.

For pretrained embeddings there were several options:

1. Compact embeddings, trained on literature and news, from Navec library
2. CBOW embeddings, trained on russian Wikipedia corpora, from RusVectors
3. Fasttext embeddings, trained on Russian documents which were stratified by regions (GEOWAC), from RusVectors

The first two options have limited dictionary of words, which became the main drawback with these embeddings: literature and news usually have minimum number of misprints or paraphrased words, but usual user queries have them a lot. After query tokenization and lemmatization, only 59% embeddings from vocabulary of text dataset were obtained using Navec and 70% for embedding from RusVectors.

On the other hand, fasttext embeddings can handle OOV (out-of-vocabulary) problem: also, they preserve distance, if word has one or two letters misprint and can work without lemmatization. Hense, for a benchmark model, fast text embeddings were chosen.

For the purpose of tokenization and meaningful word embeddings, users queries were cleared from any punctuation and numbers, as usually numbers shows only the model of the product (like user query “iphone 10”), which is not necessary for the classification.

The class distribution in the text dataset is imbalanced: 81% percent of dataset instances are not visual commercial, and 19% are visual commercial. In this case, accuracy can’t be used as a benchmark metric. As for this experiment, the importance for classes is equal, so AUC ROC was used to determine the best model and the impact of noisy detection methods.

To determine the impact of noisy labels, the set of models with different hyperparameters (different number of layers in LSTM and number of neurons of the last fully-connected layers) were trained, validated and tested on a ground truth labels (assessors answer). Then, the model with the biggest AUC ROC score were selected.

Num of layers in LSTM	**Neurons of last FC layer**	**AUC ROC score**
1	128	0.75
2	128	0.83
3	128	0.82
4	128	0.68
1	256	0.77
2	256	0.78
3	256	0.62
4	256	0.65

Table 13: comparison of the models with different hyperparameters for a benchmark in text dataset

The model with 2 number of layers in LSTM and 128 neurons in the last fully-connected layer was chosen, with AUC ROC score of 0.83. This model will be used as an upper-bound, as this or greater results model should be obtained, if labels were clean. The classification report for this model on a ground-truth labels can be shown on the next table (threshold were selected by maximizing geometric mean of sensivity and specificity).

Label	Precision	Recall	f1-score	Num of instances
Not visual commercial	0.9377	0.82	0.8751	1174
Visual commercial	0.6143	0.84	0.7096	400

Table 14: classification report for a ground-truth model on a text dataset

Next, the model with the best hyperparameters were trained and validated on noisy labels (tolokers answers) and tested on ground truth labels. This model will be used as the lower-bound, as this model wasn't model to handle label noise. This model got AUC ROC score of 0.74. The classification report for this model on a ground-truth labels can be shown on the next table.

Label	Precision	Recall	f1-score	Num of instances
Not visual commercial	0.8516	0.9727	0.9082	1174
Visual commercial	0.8627	0.5025	0.6351	400

Table 15: classification report for a model without any label noise correction for a text dataset

Hence, in this experiment the upper-bound for ROC AUC score for the text dataset is 0.83 and the lower-bound is 0.74.

4.2 Mixup

Method description

In 2018 mixup method was introduced in paper [82]. The method trains a neural network model on convex combinations of pairs of instances and their labels. Mixup method is a variation of a data augmentation techniques, as it tries to create bigger and more deverse training dataset, so neural model doesn't memorize whole training dataset, but tries to find common patterns. The augmentation creates using formulas:

$$\tilde{x} = \lambda x_i + (1 - \lambda)x_j \text{ (4.1)}$$

$$\tilde{y} = \lambda y_i + (1 - \lambda)y_j \text{ (4.2)}$$

Where x_i, x_j are input vectors and y_i, y_j are one-hot-vector label encodings. $\lambda \epsilon [0,1]$. This formula tries to use the fact, that linear interpolations of feature vectors should give linear interpolations of corresponding vectors. λ parameter is sampled from beta distribution on each batch.

Despite this method is quite simple, paper [82] shows performance increase on state-of-the-art datasets, like CIFAR-10, CIFAR-100 and ImageNet-2012, also it increases the robustness of the model to noisy labels or adversarial examples. Another advantage of this method is that it needs only few lines of code and computationally cheap.

Pictures dataset

The benchmark model with mixup got Accuracy score of 0.6632. The classification report for this model on a ground-truth labels can be shown on the next table.

Label	Precision	Recall	f1-score	Num of instances
Good quality	0.6717	0.7839	0.7235	509
Ok quality	0.6451	0.5558	0.5951	484
Bad quality	0.7031	0.6743	0.6884	439

Table 16: classification report for a model using mixup method for a picture dataset

Text dataset

For a text dataset, mixup modification was used: the mixup combines λ * N words from the prefix of the first data sample and (1- λ) * M from the postfix of the second data sample.

The benchmark model with mixup got AUC ROC score of 0.765. The classification report for this model on a ground-truth labels can be shown on the next table.

Label	Precision	Recall	f1-score	Num of instances
Not visual commercial	0.866	0.968	0.914	1174
Visual commercial	0.858	0.5625	0.676	400

Table 17: classification report for a model using mixup method for a text dataset

4.3 Co-teaching

Method description

In 2018 Co-teaching method was introduced in paper [83]. The method trains two deep neural networks simultaneously in a way that they teach each other on every training batch. It uses paradigm, that neural network firstly trains on easy examples and then goes to the hard one, as training process continues. Thus, looking at the training instances with the smallest loss, two model can learn to each other, making them more robust to the label noise.

The method uses following algorithm:

1. Each network feeds training data and selects some data instances with the smallest loss, which are considered to be clean.
2. networks share their selections to each other.
3. Network back propagates the data which were selected.

At the beginning of the training, the sample size of data instances, which are consider clean, is big and decreases as the training continues. The reason for this is the mentioned paradigm, that easy instances will be learnt at the beginning of the training, so the sample size can be big, but in order to avoid overfitting, this sample size decreased, so there is less chance that misslabeled instance will be added to the sample.

The usage of two neural networks helps as two networks will have different decision boundaries and different abilities to learn, so they have different abilities to filter label noise.

One of the biggest advantages of Co-teaching is that it can be used on a noisy dataset with heavy label noise, as they tested in the paper [83]. Also, it doesn’t require clean dataset.

Pictures dataset

The benchmark model with Co-teaching got Accuracy score of 0.663. The classification report for this model on a ground-truth labels can be shown on the next table.

Label	Precision	Recall	f1-score	Num of instances
Good quality	0.665	0.735	0.698	509
Ok quality	0.652	0.489	0.559	484
Bad quality	0.611	0.706	0.655	439

Table 18: classification report for a model using Co-teaching method for a picture dataset

Text dataset

The benchmark model with Co-teaching got AUC ROC score of 0.746. The classification report for this model on a ground-truth labels can be shown on the next table.

Label	Precision	Recall	f1-score	Num of instances
Not visual commercial	0.857	0.960	0.905	1174
Visual commercial	0.819	0.532	0.645	400

Table 19: classification report for a model using Co-teaching method for a text dataset

4.4 Confident learning

Method description

In 2019 confident learning method (CL) was introduced in paper [84]. CL is working on three principles:

1. prune noisy data (instead of fixing labels or modifying the loss function),
2. count to estimate noise (instead of learning weights during training),
3. rank data instances to train with confidence (as opposed to weighting by exact probabilities)

The algorithm consists of three steps:

1. Estimate the joint distribution of given, noisy labels and latent (unknown) uncorrupted labels to fully characterize class-conditional label noise
2. Find and prune noisy instances which considered less confident

3. Train with errors removed, re-weighting examples by the estimated latent prior

Unlike other common noisy labels correction methods, CL doesn't require hyperparameters, as it uses cross-validation to obtain predicted probabilities out-of-sample. CL also estimates the joint distribution of noisy and true labels, which can be useful to debug where markup problems usually occur. Also, it is non-iterative, can be used on any model and does not require clean dataset.

Pictures dataset

The benchmark model with CL got Accuracy score of . The classification report for this model on a ground-truth labels can be shown on the next table.

Label	Precision	Recall	f1-score	Num of instances
Good quality	0.685	0.734	0.698	509
Ok quality	0.673	0.529	0.580	484
Bad quality	0.641	0.706	0.698	439

Table 20: classification report for a model using confident learning method for a picture dataset

Text dataset

The benchmark model with CL got AUC ROC score of . The classification report for this model on a ground-truth labels can be shown on the next table.

Label	Precision	Recall	f1-score	Num of instances
Not visual commercial	0.885	0.957	0.919	1174
Visual commercial	0.835	0.635	0.721	400

Table 21: classification report for a model using confident learning method for a text dataset

Chapter 5

Experimental assessment of the quality of the proposed methods

The experimental methods, which was chosen in the work, are the state of art methods for working with noisy labels, which was developed in recent years, but every method has its own advantages and restrictions. The comparison of the chosen methods can be shown in the table:

Methods	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**
Mixup	●	●	●	●	■	●
Co-teaching	●	●	■	●	●	▲
Confident learning	●	●	■	●	●	▲

Table 22: Characteristics comparison of the chosen methods in the papers

Assigned marks:

● - method fully supported this property

▲ - method supported this property partionaly

■ - method doesn’t hold this property.

Columns description:

(P1) Flexibility: Method supports any deep neural network architecture.

(P2) No Pre-training: Method doesn’t need any pretraining steps.

(P3) Full Exploration: Method uses all training instances, without removing suspicious example from the training.

(P4) No clean dataset: Method does not need clean dataset to start working.

(P5) Heavy Noise tolerant: Method can work with heavy noise datasets

(P6) Tolerant to a different type of Noise: method can work with different type of noise (instance-dependent or not-independent).

The comparison of methods for a text and pictures datasets can be shown in the tables:

Methods	**Accuracy**	**Average Precision**	**Average Recall**	**Average F1-score**
Model learned on ground-truth label	0.68	0.72	0.63	0.67
Confident learning	0.67	0.69	0.66	0.66
Mixup	0.66	0.66	0.67	0.66
Co-teaching	0.66	0.65	0.65	0.65
Model without any label noise correction	0.64	0.63	0.64	0.63

Table 23: Experiment results comparison on a picture dataset

Methods	**ROC AUC score**	**Average Precision**	**Average Recall**	**Average F1-score**
Model learned on ground-truth label	0.83	0.82	0.83	0.83
Confident learning	0.79	0.86	0.80	0.82
Mixup	0.76	0.86	0.76	0.79
Co-teaching	0.75	0.86	0.76	0.79
Model without any label noise correction	0.74	0.85	0.73	0.77

Table 24: Experiment results comparison on a text dataset

Chapter 6

Conclusion

In this paper, the main state-of-the-art robust methods of working with noisy labels were reviewed. The problem of noisy labels has a long research story, starting with classical machine learning methods and continuing with modern neural network methods. Various new methods try to solve this problem in different ways, but still there is no universal way that can solve all possible problems of noisy labels.

Based on the development in the neural network in general and the growing interest in more complex problems in texts and pictures, researchers and industry will need bigger and more complex datasets. As was reviewed in the paper, neither experts, nor main crowdsourcing platforms can guarantee a ground-truth quality for the dataset markup for a various reasons, and this reasons are aggravated by increasing markup tasks diversity and complexity. To model new types of label noise new robust and accurate method needed, which also develops how they treat label noise, moving from instance-independent noise to an instance-dependent.

The experiment carried out in this work on real picture and text datasets showed the effectiveness of new methods in practice, apparently allowing to increase the accuracy and quality of the models.

References

[1] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, Jae-Gil Lee Learning from Noisy Labels with Deep Neural Networks: A Survey 2021

[2] Davood Karimi, Haoran Dou, Simon K. Warfield, and Ali Gholipour Deep learning with noisy labels: exploring techniques and remedies in medical image analysis 2020

[3] Nagarajan Natarajan, Inderjit S. Dhillon, Pradeep Ravikumar Learning with Noisy Labels 2013

[4] Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, Li-Jia Li Learning from Noisy Labels with Distillation 2017

[5] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study," Artificial Intelligence Review 2004

[6] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," 2017

[7] W. Mason and S. Suri, "Conducting behavioral research on amazon's mechanical turk," Behavior Research Methods 2012

[8] C. Scott, G. Blanchard, and G. Handy, "Classification with asymmetric label noise: Consistency and maximal denoising," 2013

[9] B. Frenay and M. Verleysen, "Classification in the presence of label noise: A survey," 2013

[10] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, "Learning from massive noisy labeled data for image classification," 2015

[11] H. Song, M. Kim, and J.-G. Lee, "SELFIE: Refurbishing unclean samples for robust deep learning," 2019

[12] B. Frenay and M. Verleysen, "Classification in the presence of label ´ noise: A survey," 2013

[13] Keren Gu, Xander Masotto, Vandana Bachani, Balaji Lakshminarayanan, Jack Nikodem, Dong Yin. “An Instance-Dependent Simulation Framework for Learning with Label Noise” 2021 ·

[14] B. Frenay and M. Verleysen, “Classification in the presence of label ´ noise: a survey,”, 2013.

[15] V. Hodge and J. Austin, “A survey of outlier detection methodologies,”, 2004.

[16] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,”, 2009.

[17] D. F. Nettleton, A. Orriols-Puig, and A. Fornells, “A study of the effect of different types of noise on the precision of supervised learning techniques,”, 2010.

[18] Y. Bi and D. R. Jeske, “The efficiency of logistic regression compared to normal discriminant analysis under class-conditional classification noise,”, 2010.

[19] J.-w. Sun, F.-y. Zhao, C.-j. Wang, and S.-f. Chen, “Identifying and correcting mislabeled training instances,”, 2007.

[20] D. Gamberger, N. Lavrac, and S. D ˇ zeroski, “Noise elimination in ˇ inductive concept learning: A case study in medical diagnosis,” 1996.

[21] J. Thongkam, G. Xu, Y. Zhang, and F. Huang, “Support vector machine for outlier detection in breast cancer survivability prediction,”, 2008

[22] A. L. Miranda, L. P. Garcia, A. C. Carvalho, and A. C. Lorena, “Use of classification algorithms in noise detection and elimination,”, 2009

[23] B. Sluban, D. Gamberger, and N. Lavrac, “Advances in class noise detection,”, 2010,

[24] H. Berthelsen and B. Megyesi, “Ensemble of classifiers for noise detection in pos tagged corpora,” , 2000

[25] F. Muhlenbach, S. Lallich, and D. A. Zighed, “Identifying and handling mislabelled instances,”, 2004.

[26] B. Dasarathy, “Nearest neighbor (NN) norms: nn pattern classification techniques.”, 1991.

[27] S. Verbaeten and A. Van Assche, “Ensemble methods for noise elimination in classification problems,” 2003

[28] V. Wheway, “Using boosting to detect noisy data,” 2001

[29] C. Zhang, C. Wu, E. Blanzieri, Y. Zhou, Y. Wang, W. Du, and Y. Liang, “Methods for labeling error detection in microarrays based on the effect of data perturbation on the regression model,”, 2009.

[30] A. Malossini, E. Blanzieri, and R. T. Ng, “Detecting potential labeling errors in microarrays by data perturbation,”, 2006.

[31] C. J. Perez, F. J. Giron, J. Martin, M. Ruiz, and C. Rojano, “Misclassified multinomial data: a bayesian approach, 2007

[32] E. Eskin, “Detecting errors within a corpus using anomaly detection”, 2000.

[33] C. Bouveyron and S. Girard, “Robust supervised classification with mixture models: Learning from data with uncertain labels,” , 2009.

[34] R. Rosales, G. Fung, and W. Tong, “Automatic discrimination of mislabeled training points for large margin classifiers,” 2009

[35] L. Xu, K. Crammer, and D. Schuurmans, “Robust support vector machine training via convex outlier ablation,” 2006

[36] C. Domingo and O. Watanabe, “Madaboost: A modification of adaboost,” 2000

[37] N. C. Oza, “Boosting with averaged weight vectors, 2003

[38] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, “Understanding deep learning requires rethinking generalization,”, 2016.

[39] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio et al., “A closer look at memorization in deep networks,” , 2017

[40] F. Wang, L. Chen, C. Li, S. Huang, Y. Chen, C. Qian, and C. Change Loy, “The devil of face recognition is in the noise” 2018

[41] D. Rolnick, A. Veit, S. Belongie, and N. Shavit, “Deep learning is robust to massive label noise,”, 2017

[42] X. Yu, T. Liu, M. Gong, K. Zhang, and D. Tao, “Transfer learning with label noise,”, 2017.

[43] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, “Universal adversarial perturbations,”, 2017

[44] J. Speth and E. M. Hand, “Automated label noise identification for facial attribute recognition,”, 2019

[45] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie, “Learning from noisy large-scale datasets with minimal supervision,”, 2017

[46] P. Ostyakov, E. Logacheva, R. Suvorov, V. Aliev, G. Sterkin, O. Khomenko, and S. I. Nikolenko, “Label denoising with large ensembles of heterogeneous neural networks,”, 2018

[47] K.-H. Lee, X. He, L. Zhang, and L. Yang, “Cleannet: Transfer learning for scalable image classifier training with label noise,”, 2018

[48] J. Han, P. Luo, and X. Wang, “Deep self-learning from noisy labels,”, 2019

[49] C. G. Northcutt, T. Wu, and I. L. Chuang, “Learning with confident examples: Rank pruning for robust classification with noisy labels,”, 2017

[50] J. Han, P. Luo, and X. Wang, “Deep self-learning from noisy labels,”, 2019

[51] H. Zhou, J. Sun, Y. Yacoob, and D. W. Jacobs, “Label denoising adversarial network (ldan) for inverse lighting of face images,” arXiv preprint arXiv:1709.01993, 2017

[52] F. Chiaroni, M. Rahal, N. Hueber, and F. Dufaux, “Hallucinating a cleanly labeled augmented dataset from a noisy labeled dataset using gans,” 2019

[53] S. Sukhbaatar, J. Bruna, M. Paluri, L. Bourdev, and R. Fergus, “Training convolutional networks with noisy labels, 2014

[54] K. K. Thekumparampil, A. Khetan, Z. Lin, and S. Oh, “Robustness of conditional gans to noisy labels, 2018

[55] R. Tanno, A. Saeedi, S. Sankaranarayanan, D. C. Alexander, and N. Silberman, “Learning from noisy labels by regularized estimation of annotator confusion,”, 2019

[56] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, “Learning from massive noisy labeled data for image classification,”, 2015

[57] A. Vahdat, “Toward robustness against label noise in training deep discriminative neural networks,”, 2017

[58] A. Ghosh, H. Kumar, and P. Sastry, “Robust loss functions under label noise for deep neural networks, 2017.

[59] Z. Zhang and M. Sabuncu, “Generalized cross entropy loss for training deep neural networks with noisy labels, 2018

[60] X. Wang, E. Kodirov, Y. Hua, and N. M. Robertson, “Improving mae against cce under label noise”, 2019

[61] S. Thulasidasan, T. Bhattacharya, J. Bilmes, G. Chennupati, and J. Mohd-Yusof, “Combating label noise in deep learning using abstention, 2019.

[62] A. Rusiecki, “Trimmed robust loss function for training deep neural networks with label noise,” , 2019

[63] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu, “Making deep neural networks robust to label noise: A loss correction approach,”, 2017.

[64] M. Ren, W. Zeng, B. Yang, and R. Urtasun, “Learning to reweight examples for robust deep learning,”, 2018

[65] X. Wang, Y. Hua, E. Kodirov, and N. Robertson, “Emphasis regularisation by gradient rescaling for training deep neural networks with noisy labels,”, 2019.

[66] B. Han, G. Niu, J. Yao, X. Yu, M. Xu, I. Tsang, and M. Sugiyama, “Pumpout: A meta approach for robustly training deep neural networks with noisy labels,”, 2018.

[67] Y. Shen and S. Sanghavi, “Learning with bad training data via iterative trimmed loss minimization,”, 2019

[68] J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, and D. Meng, “Metaweight-net: Learning an explicit mapping for sample weighting,”, 2019.

[69] K. Lee, S. Yun, K. Lee, H. Lee, B. Li, and J. Shin, “Robust inference via generative classifiers for handling noisy labels, 2019

[70] S. Azadi, J. Feng, S. Jegelka, and T. Darrell, “Auxiliary image regularization for deep cnns with noisy labels,”, 2015

[71] C. Li, C. Zhang, K. Ding, G. Li, J. Cheng, and H. Lu, “Bundlenet: Learning with noisy label via sample correlations,”, 2017

[72] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, “Curriculum learning,”, 2009

[73]] S. Guo, W. Huang, H. Zhang, C. Zhuang, D. Dong, M. R. Scott, and D. Huang, “Curriculumnet: Weakly supervised learning from largescale web images” 2018

[74]] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L.-J. Li, “Learning from noisy labels with distillation,”, 2017

[75] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, “Training deep neural networks on noisy labels with bootstrapping,”, 2014

[76] E. Malach and S. Shalev-Shwartz, “Decoupling” when to update” from” how to update”,”, 2017

[77] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, “Co-teaching: Robust training of deep neural networks with extremely noisy labels,”, 2018

[78] X. Yu, B. Han, J. Yao, G. Niu, I. Tsang, and M. Sugiyama, “How does disagreement help generalization against label corruption”, 2019

[79] P. Chen, B. Liao, G. Chen, and S. Zhang, “Understanding and utilizing deep neural networks trained with noisy labels,”, 2019

[80] S. Song, K. Chaudhuri, and A. Sarwate, “Learning from data with heterogeneous noise using sgd,”, 2015

[81] S. Sukhbaatar and R. Fergus, “Learning from noisy labels with deep neural networks,”, 2014

[82] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, David Lopez-Paz, “Mixup: Beyond Empirical Risk Minimization”, 2018

[83] Curtis G. Northcutt, Lu Jiang, Isaac L. Chuang, “Confident Learning: Estimating Uncertainty in Dataset Labels”, 2021

[84] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, Masashi Sugiyama , “Co-teaching: Robust Training of Deep Neural Networks with Extremely Noisy Labels”, 2018

List of pictures

List of tables