ActiveClean: Interactive Data Cleaning For Statistical Modeling

Safkat Islam
Carolyn Zhang
CS 590
Outline

- Biggest Takeaways, Strengths, and Weaknesses
- Background
- System Architecture
- Updating the Model
- Detecting Dirty Data
- Selecting Which Records to Clean
- Experimental Setup and Results
- Conclusion
- Demo
Biggest Takeaways

- ActiveClean is a framework for the iterative cleaning of a dataset for statistical modelling while maintaining convergence for convex loss problems.

- The authors formulate the iterative cleaning problem as a convex optimization problem, as a result, they can use stochastic gradient descent in ActiveClean.

- Prioritizing cleaning of data points that have a higher probability of affecting the results of the model can increase predictive accuracy of the model while cleaning the same amount of data.
Strengths

- Experimental setup
  - Real applications
  - Against four other methods
- Batch cleaning increases efficiency (in a similar manner as Sample-Clean)
- Detector works with other commonly used classes of rules
  - Integrity constraints
  - CFDs
  - Matching dependencies
  - Adaptive error detection

Weaknesses

- Initializing the model on the dirty dataset
  - Data could be cleaned of common inconsistencies
- Approach to sampling
  - More sensitive to outliers in dataset due to approach to SGD
- Modification of the stochastic gradient descent algorithm
  - Increases the time required for the gradient calculation with each iteration.
- Nonlinear gradients
  - No isolation of features
  - Use Taylor series approximation instead
Background

- Iterative data cleaning is:
  - Cleaning subsets of data
  - Evaluating preliminary results
  - Cleaning more data as necessary

- Assumptions
  - Approaches a class of analytics problems that can be expressed as the minimization of convex loss functions
  - Data cleaning operations are applied record-by-record (alternative: find-and-replace where operations are applied on the full dirty dataset as a set-of-records)
System Architecture

- **Initialization** - train model on dirty data
- **Sampler** - selects a batch of data from dirty data in a randomized way but can assign probabilities
- **Cleaner** - user-specified function
- **Updater** - updates the model based on cleaned data using a SGD step
- **Detector** (optional) - identifies records that are more likely to be dirty
- **Estimator** (optional) - uses previously cleaned data to estimate the effect of cleaning a record on the model
System Architecture (Block Diagram)
Model Update Algorithm

1. Take a sample of data $S$ from $R_{dirty}$
2. Calculate the gradient over the sample of newly clean data and call the result $g_S(\theta^{(t)})$
3. Calculate the average gradient over all of the already clean records in $R_{clean} = R - R_{dirty}$, and call the result $g_C(\theta^{(t)})$
4. Apply the following update rule, which is a weighted average of the gradient on the already clean records and newly cleaned records:
   \[
   \theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma \cdot \left( \frac{|R_{dirty}|}{|R|} \cdot g_S(\theta^{(t)}) + \frac{|R_{clean}|}{|R|} \cdot g_C(\theta^{(t)}) \right)
   \]
5. Append the newly cleaned records to set of previously clean records $R_{clean} = R_{clean} \cup S$
Updating the Model

- Modification of stochastic gradient descent algorithm
- **Stochastic Gradient Descent**: iteratively minimizes the objective function
  - Convex loss function: function measures the accuracy (predictive error) of the statistical model
  - Geometric interpretation:

\[
\theta_{\text{new}} \leftarrow \theta^{(d)} - \gamma \cdot \nabla \phi(\theta^{(d)})
\]

- **ActiveClean**: estimates the gradient by averaging the gradient of a sample of dirty data and the gradient of the cleaned data

\[
g(\theta) = \left| \frac{R_{\text{clean}}}{R} \right| \cdot g_C(\theta) + \left| \frac{R_{\text{dirty}}}{R} \right| \cdot g_S(\theta)
\]
Detecting Dirty Data

- **Detector returns:**
  - Whether a record is dirty
  - And if it is dirty, which attributes have errors

- **Requirement:** there is an algorithm to enumerate the set of records that violate at least one rule
  - Let the set of clean data be the union between the set of clean data and set of records that satisfies all of the rules
  - The set of dirty data is the set of records that violate at least one rule
  - Apply the model updating algorithm
Adaptive Detection

- Dirty data is categorized into $u$ classes (corruption) using a multi-class classifier
  - These classes may not align with the features but every record is classified with at most one corruption category
- The repair step cleans the data and reports which of the $u$ classes is associated with this record
- **Algorithm for updating workflow:**
  - $R_{\text{clean}}$ is previously cleaned data and $U_{\text{clean}}$ is a set of labels for each record indicating error class and dirty/not-dirty
  - Train classifier to predict the label (Train)
  - Apply classifier to dirty data (Predict)
  - For all records that are predicted to be clean, remove them from $R_{\text{dirty}}$ and append to $R_{\text{clean}}$
  - Apply model updating algorithm
Selecting Which Records to Clean

- **Optimal sampling problem:**
  - Want to minimize the variance of the sampled gradient from the gradient of the dirty subset of the dataset
  - Approach: use the detector to estimate the cleaned values

- **Estimator:**
  - Estimates the clean gradient using the dirty gradient and the results from the Detector
  - Linear approximation of gradient: uses average change of each feature value

- **Limitation:** couples impact of features and model results
Experimental Setup

- **Comparisons** to Naive-Mix, Naive-Sampling, Active Learning, and Oracle

- **Metrics:**
  - **Model error** - the distance between the trained model and the true model if all data were cleaned
  - **Test error** - the prediction accuracy of the model on a held out set of clean data

- **Comparison Axes:**
  - The sampling procedure to pick the next set of records to clean
  - The model update procedure to incorporate the cleaned sample

- **Scenarios:**
  - Movie prediction from IMDB and Yahoo databases
  - ProPublica’s Dollars for Docs (DfD) database for medical donations
  - Simulated ML pipeline (AMPLab’s Keystone MOL and Google’s Tensor Flow) to classify handwritten numbers
  - Simulated scenarios for predicting income bracket of an adult
  - Simulated scenario for predicting onset of a seizure
Experimental Setup (Continued)

- **Simulated Scenarios:**
  - **Goal** - understand which types of data corruption are amenable to data cleaning versus robust statistical techniques
  - **Four schemes:**
    - Full data cleaning
    - Baseline of no cleaning
    - Discarding dirty data
    - Robust logistic regression
  - Corrupted (both randomly and systematically) 5% of training examples in each dataset
  - **Results:**
    - Robust method works well on random corruption but falters on systematic corruption
    - Discarding dirty data also works well for random corruption
Experimental Results

- **Movies** - ActiveClean has superior convergence
  - Update algorithm correctly incorporates the raw data with the cleaned samples leaded to smaller sensitivity to sampling error
  - Selects records that are more likely to be dirty and that will mostly improve the model

- **DfD** - ActiveClean can effectively identify and exploit bias of systematic errors to quickly converge to the true clean model
  - Naive-Mix (most commonly used approach) was almost completely ineffective

- **ML Pipeline** - ActiveClean makes more progress towards the clean model with a smaller number of examples cleaned
Experimental Results (IMDB)

(a) IMDB Model Error

(b) IMDB Test Error
Experimental Results (DfD)

(c) DfD Model Error

(d) DfD False Negatives

- Dirty
- AC
- O
- NS
- AL
- NM
Experimental Results (ML Pipeline)

MNIST Block Removal 5x5

MNIST Fuzzy

# Images Cleaned

Model Error %

Dirty  AC  O  NS  AL

# Images Cleaned

Model Error %
Experimental Results (Continued)

- **Simulated Scenarios:**
  - ActiveClean is consistently better than Active Learning because it is a composable framework that supports different instantiation of detection and prioritization modules but still guarantees convergence.
  - Naive-Mix is an unreliable methodology that lacks convergence guarantees and has lower efficiency than the other methodologies because it considers the entire dirty data.
  - Naive-Sampling does not use the dirty data and its error is governed by the sample size.
    - It outperforms ActiveClean when corruptions are severe or when initialization with the dirty model is inaccurate.
  - As corruption becomes more random, the classifier becomes increasingly erroneous.
    - Break-even point at about 50%.
    - Beyond that, classifier is imperfect and misclassifies some data points incorrectly as cleaned.
Experimental Results (Simulated Scenarios)

(a) Adult

(b) EEG

Model Error % vs. # Records Cleaned

- Dirty
- AC
- AL
- NM
- NM+D
Conclusion

- **Given:** dirty dataset, convex model, general data cleaning procedure
- **Problem addressed:** cleaning data before user-specified modelling while preserving convergence
- **ActiveClean** improves the efficiency of cleaning in comparison to other state of the art methods
  - Cleans fewer data records to achieve a high predictive accuracy
- **ActiveClean** is especially useful for the cleaning of systematic corruption
- **Extensions:**
  - Broaden scope of cleaning: look at errors in the dirty records that, if fixed, cause errors in the cleaned dataset
  - Relax the requirement for convex models
Demo:

http://automation.berkeley.edu/activecleandeemo
Thanks!
Any questions?