1.Why Phrase Mining? 1) Unigrams vs. phrases −−− Unigrams (single words): ambiguous, difficult to interpret; Phrase: A natural, meaningful, unambiguous semantic unit 2) Mining semantically meaningful phrases ---- Transform text data from word granularity to phrase granularity; Enhance the power and efficiency at manipulating unstructured data using database technology

2. Why not use NLP Methods? 1)High annotation cost, not scalable to a new language, a new domain or genre 2)May not fit domain-specific, dynamic, emerging applications 3) Advantage of Data mining -- General principle: Fully exploit information redundancy and data- driven criteria to determine phrase boundaries and salience

3. Phrase Mining Approaches

Strategy 1: Simultaneously Inferring Phrases and Topics *Generate bag-of-words -> generate sequence of tokens*; *High model complexity: Tends to overfitting; High inference cost: Slow*

Bigram Topic Model Probabilistic generative model that conditions on previous word and topic when drawing next word **Topical N-Grams (TNG)** Probabilistic model that generates words in textual order; Create n-grams by concatenating successive bigrams (a generalization of Bigram Topic Model) **Phrase-Discovering LDA (PDLDA)** Viewing each sentence as a time-series of words, PDLDA posits that the generative parameter (topic) changes periodically; Each word is drawn based on previous m words (context) and current phrase topic

Strategy 2: Post Topic Modeling Phrase Construction *Post bag-of-words model inference, visualize topics with n-grams; Can be fast; generally high-quality topics and phrases*

TurboTopics -- Phrase construction as a post-processing step to Latent Dirichlet Allocation Perform Latent Dirichlet Allocation on corpus to assign each token a topic label; Merge adjacent unigrams with the same topic label by a distribution-free permutation test on arbitrary-length back-off model; End recursive merging when all significant adjacent unigrams have been merged **KERT** – Phrase construction as a post-processing step to Latent Dirichlet Allocation Perform frequent pattern mining on each topic; Perform phrase ranking based on four different criteria

Framework of KERT 1. Run bag-of-words model inference and assign topic label to each token 2. Extract candidate keyphrases(Frequent pattern mining) within each topic 3. Rank the keyphrases in each topic (Popularity; Discriminativeness; Concordance; Completeness)

Strategy 3: First Phrase Mining then Topic Modeling (ToPMine)

Prior bag-of-words model inference, mine phrases and impose on the bag-of-words model; Can be fast; generally high-quality topics and phrases **TopMine** 1) First phrase construction, then topic mining; Contrast with KERT: topic modeling, then phrase mining 2) With Strategy 2, tokens in the same phrase may be assigned to different topics

Framework of ToPMine 1) Perform frequent contiguous pattern mining to extract candidate phrases and their counts; 2)Perform agglomerative merging of adjacent unigrams as guided by a significance score—This segments each document into a "bag-of-phrases"; 3)The newly formed bag-of-phrases are passed as input to PhraseLDA, an extension of LDA, that constrains all words in a phrase to each sharing the same latent topic

4. SegPhrase

ClassPhrase*: Frequent pattern mining, feature extraction, classification*

Phase1: Build a candidate phrases set by frequent pattern mining Mining frequent k-grams (k is typically small, e.g. 6 in our experiments); Popularity measured by raw frequent words and phrases mined from the corpus **Phase2**: Feature Extraction Concordance **Phase2**: Feature Extraction Informativeness (1)Quality phrases typically start and end with a non-stopword;2) Use average IDF over words in the phrase to measure the semantics; 3)Usually, the probabilities of a quality phrase in quotes, brackets, or connected by dash should be higher (punctuations information) 4) *OR Incorporate features using some NLP techniques, such as POS tagging, chunking, and semantic parsing* **Phase3:** 1) Limited Training(Labels: Whether a phrase is a quality one or not); 2) Random Forest as our classifier

SegPhrase*: Phrasal segmentation and phrase quality estimation* 1)Phrasal segmentation can tell which phrase is more appropriate 2) Rectified phrase frequency (expected influence) **Phase1**: Partition a sequence of word by maximizing the likelihood (considering phrase quality score, probability in corpus and length penalty) **Phase2**: Filter out phrases with low rectified frequency (Bad phrases are expected to rarely occur in the segmentation results)

SegPhrase+*: One more round to enhance mined phrase quality (P;D;C;C)* Feedback: Using rectified frequency, re-compute those features previously computing based on raw frequency

Advantage: Integrating phrase mining with phrasal segmentation; Requires only limited training or distant training; Generates high-quality phrases, close to human judgement; Linearly scalable on time and space

5. Entity Recognition

1) **Traditional named entity recognition systems are designed for major types (e.g., PER, LOC, ORG) and general domains (e.g., news):** Require additional steps to adapt to new domains/types; Expensive human labor on annotation; Unsatisfying agreement due to various granularity levels and scopes of types **2) Entities obtained by entity linking techniques have limited coverage and freshness** **3) A new approach:** ClusType: Entity Recognition and Typing by Relation Phrase-Based Clustering : Recognizing entity mentions of target types with minimal/no human supervision and with no requirement that entities can be found in a KB (distant supervision)

Weak supervision: relies on manually selected seed entities in applying pattern-based bootstrapping methods or label propagation methods to identify more entities (Both assume seeds are unambiguous and sufficiently frequent; requires careful seed selection by human) **Distant supervision**: leverages entity information in KBs to reduce human supervision (cont.)

Challenge & Solution 1) Domain-agnostic phrase mining algorithm: Extracts candidate entity mentions with minimal linguistic assumption -> address domain restriction (E.g., part-of-speech (POS) tagging << semantic parsing); 2) Do not simply merge entity mentions with identical surface names (Model each mention based on its surface name and context, in a scalable way) -> address name ambiguity; 3) Mine relation phrase co-occurring with entity mentions; infer synonymous relation phrases (Helps form connecting bridges among entities that do not share identical context, but share synonymous relation phrases) -> address context sparsity

ClusType Framework 1) Phrase Segmentation and Heterogeneous Graph Construction: POS-constrained phrase segmentation for mining candidate entity mentions and relation phrases, simultaneously; Construct a heterogeneous graph to represent available information in a unified form; 2) Mutual Enhancement of Type Propagation and Relation Phrase Clustering: With the constructed graph, formulate a graph-based semi-supervised learning of two tasks jointly: Type propagation on heterogeneous graph <=> Multi-view relation phrase clustering => Derived entity argument types serve as good feature for clustering relation phrases; Propagate type information among entities bridges via synonymous relation phrases; Mutually enhancing each other; leads to quality recognition of unlinkable entity mentions

General Overview

Step 1 Candidate Generation: Perform phrase mining on a POS-tagged corpus to extract candidate entity mentions and relation phrases ------- An efficient phrase mining algorithm incorporating both corpus-level statistics and syntactic constraints 1)Global significance score: Filter low-quality candidates; generic POS tag patterns: remove phrases with improper syntactic structure 2) By extending TopMine, the algorithm partitions corpus into segments which meet both significance threshold and POS patterns --> candidate entity mentions & relation phrases

Step 2 Construction of Heterogeneous Graphs: Construct a heterogeneous graph to encode our insights on modeling the type for each entity mention; Collect seed entity mentions as labels by linking extracted mentions to the KB --------With three types of objects extracted from corpus: candidate entity mentions, entity surface names, and relation phrases (Entity Mention-Surface Name Subgraph; Entity Name -Relation Phrase Subgraph; Mention Correlation Subgraph)

Hypothesis 1 (Entity-Relation Co-occurrences): If surface name c often appears as the left (right) argument of relation phrase p, then c's type indicator tends to be similar to the corresponding type indicator in p's type signature. **Hypothesis 2** (Mention correlation): If there exists a strong correlation (i.e., within sentence, common neighbor mentions) between two candidate mentions that share the same name, then their type indicators tend to be similar.

Step 3 Relation Phrase Clustering: Estimate type indicator for unlinkable candidate mentions with the proposed type propagation integrated with relation phrase clustering on the constructed graph ------- Existing work on relation phrase clustering utilizes strings; context words; entity argument to cluster synonymous relation phrases; String similarity and distribution similarity may be insufficient to resolve two relation phrases; type information is particular helpful in such case; We propose to leverage type signature of relation phrase,and propose a general relation phrase clustering method to incorporate different features

Hypothesis 3 (Type signature consistency): If two relation phrases have similar cluster memberships, the type indicators of their left and right arguments (type signature) tend to be similar, respectively. **Hypothesis 4** (Relation phrase similarity): Two relation phrases tend to have similar cluster memberships, if they have similar (1) strings; (2) context words; and (3) left and right argument type indicators)

Experiment Setting 1) Seed mention sets: < 7% extracted mentions are mapped to Freebase entities 2) Evaluation sets: manually annotate mentions of target types for subsets of the corpora 3) Evaluation metrics: Follows named entity recognition evaluation (Precision, Recall, F1) 4) Compared methods --- **Pattern**: Stanford pattern-based learning; **SemTagger**: bootstrapping method which trains contextual classifier based on seed mentions; **FIGER**: distantly-supervised sequence labeling method trained on Wiki corpus; **NNPLB**: label propagation using ReVerb assertion and seed mention; **APOLLO:** mention-level label propagation using Wiki concepts and KB entities; **ClusType-NoWm**: ignore mention correlation; **ClusType-NoClus**: conducts only type propagation; **ClusType-TwpStep**: first performs hard clustering then type propagation

6. LINE: Large-scale Information Network Embedding

Hypothesis 1) Nodes with strong ties turn to be similar (**1st order similarity** Try to capture local pairwise proximity between the nodes (vertices) in the network: Vertices that are connected with larger weight are more similar); 2) Nodes share many neighbors turn to be similar (**2nd order similarity** proximity between two vertices (u, v) is the similarity between their neighborhood network structure: Each vertex is treated as a specific context, and vertices with similar distributions over the contexts are assumed to be similar) => Well-learnt embedding should preserve both 1st order and 2nd order similarity

Algorithm LINE and Some Practical Issues 1) LINE:Train the LINE model which preserves the first-order proximity and second-order proximity separately; Then concatenate the embeddings trained by the two methods for each vertex; 2) **A more principled way** (future work): Jointly train the two objective functions **3) Implementation consideration**: Information networks can be very sparse: Reconstruct them to make them denser (Strategy: Add links between every vertex and its high order neighbors; Add second order neighbors)

Previous Work 1) Graph Embedding: Construct an affinity matrix & compute eigenvectors of the affinity matrix; Weakness: High time complexity and space complexity, and Only preserve 1st order similarity **2) Matrix Factorization:** Objective: Optimization: SGD (Stochastic Gradient Descent Alg.)**;** Weakness: Can't converge when the range of weights is very large, Only preserve 1st order similarity **3) Deep Walk**: Sample a chain of vertex by truncated random walk**;** Treat the chain as a sentence and run word2vec on this sentence; Weakness: Only preserve 2nd order similarity

7. Mining Biological Relationship by Integration of LINE with Phrase Mining - Framework 1) Construct a biological entity list --> SegPhrase / Biological vocabulary 2) Detect and extract biological entities from biological text (research papers), using SegPhrase / Maximum Matching 3) Construct a co-occurrence network between words and biological entities 4) Learn embedding vector for each entity by using a network embedding technique LINE (i.e., Run LINE-2nd to learn entity embeddings): Entities share many neighbors in the co-occurrence network tend to be similar 5) Given seed entity pair (a, b) and a query entity x, return an entity y according to their embedding vectors, so that the relation between x and y is similar to the relation between a and b.

8. LAKI: Representing Documents via Latent Keyphrase Inference

Previous Work: 1) Bag-of-Words or Bag-of-Phrases Cons: Sparse on short texts **2) Topic models [LDA]:** Each topic is a distribution over words; each document is a mixture of corpus-wide topics Cons: Difficult for human to infer topic semantics **3) Concept-based models [ESA]** Cons: Low coverage of concepts in human-curated knowledge base **4) Word/document embedding models [word2vec]** Cons: Difficult to explain what each dimension means

Framework Offline: Phrase Mining --> Quality Phrase Silhouetting(learning hierarchical Bayesian Network(DAG): Model learning + Structure learning); **Online:** Segmentation --> Document Keyphrase Inference --> Document Representation

9. TruthFinding

TruthFinder 1) Confidence of facts ↔ Trustworthiness of web sites (A fact has high confidence if it is provided by (many) trustworthy web sites; A web site is trustworthy if it provides many facts with high confidence) 2) **TruthFinder mechanism**: Initially, each web site is equally trustworthy; Based on the above four heuristics, infer fact confidence from web site trustworthiness, and then backwards; Repeat until achieving stable state **3) vs. authority-hub analysis:** Facts ↔ Authorities, Web sites ↔ Hubs 1) Linear summation cannot be used (A web site is trustable if it provides accurate facts, instead of many facts; Confidence is the probability of being true) 2) Different facts of the same object influence each other

LTM: Latent Truth Model: A Principled Probabilistic Model; Model negative claims and two-sided source quality with Bayesian regularization

vs. HITS-like Random Walk methods (e.g., TruthFinder, 3-Estimate, Invest)**:** HigherQualitySources↔MoreProbableFacts **Limitations: 1)** Quality as a single value:Precision or Accuracy **2)** In practice, some sources tend to ignore true attributes(False Negatives), while some others tend to produce false attributes (False Positives). 3) FN≠FP when there are multiple truths per entity

Model: 1) For each source k: Generate false positive rate (with strong regularization, believing most sources have low FPR); Generate its sensitivity (1-FNR) with uniform prior, indicating low FNR is more likely; **2) For each fact f:** Generate its prior truth prob, uniform prior; Generate its truth label; **3) For each claim c of fact f, generate observation of c**. If f is false, use false positive rate of source; If f is true, use sensitivity of source.

Inferring Truth: 1) MAP inference: find truth assignment that maximizes posterior probabilities Collapsed Gibbs sampling (more efficient, only sample truth, other parameters integrated out) 3) Prediction of the truth from samples of the latent truth variable (Burn-in: throw away first b samples; Thinning: Take every k sample from all the samples; Calculate expectation of the taken samples)

Incremental Prediction (LTMinc): Assuming source quality unchanged, directly utilize source quality to make prediction (very efficient); Can also rerun inference for incremental update by using previous quality counts as Bayesian priors

GTM: (A Gaussian Truth Model for Finding Truth among Numerical Data) A principled probabilistic model; Leverage two Gaussian generative processes to simulate the generation of numerical truth and claims; Source quality adapts to numerical data; Prior on truth and source quality can be easily incorporated as Bayesian priors; Efficient inference **Normalization and Outlier Detection:** 1) Leverage truth priors, e.g., median, the most frequent value, or output of any truth-finding algorithms; 2) Remove outliers by absolute error, relative error, and Gaussian confidence interval (with prior truth as mean, iteratively computed variance): Outliers can significantly shift empirical variance, so update variance after outliers are detected and try to detect outliers based on updated variance; 3) Normalize all claims to standard Gaussian N(0,1): Prevent biased estimation of source quality **Mechanism**: For each source k: Generate source quality; For each entity e: Generate its true value; For each claim c of entity e: Generate observation of c. **Inference:** Likelihood; MAP Inference of Truth; Many inference algorithms can be applied, e.g. Gibbs sampling, EM, etc.

10. Stream Mining

Data Streams 1) Features: Continuous, ordered, changing, fast, huge volume; Contrast with traditional DBMS(finite, persistent

datasets; **2) Characteristics**: Huge volumes of continuous data, possibly infinite; Fast changing and requires fast, real-time response; Data stream captures nicely our data processing needs of today; Random access is expensive: single scan algorithm (can only have one look); Store only the summary of the data seen thus far; Most stream data are at low-level and multi-dimensional in nature, needs multi-level and multi-dimensional processing

Stream Cube Architecture: 1) A tilted time frame: Different time granularities **2) Critical layers**: Minimum interest layer (m-layer: Stream data is generalized to m-layer (minimal interest layer) and "stored" to facilitate flexible drilling) & Observation layer (o-layer: Stream data should be constantly summarized and presented at the o-layer (observation layer) for constant observation) => User: watches at o-layer and occasionally needs to drill-down down to m-layer **3) Partial materialization of stream cubes** (Full materialization: too space and time consuming; No materialization: slow response at query time; Cube partial materialization: Store some pre-computed cuboids for fast online processing, **Popular-path approach**--Materializing those along the popular drilling paths / **H-tree structure** -- Space preserving, Intermediate aggregates can be computed incrementally and saved in tree nodes; Facilitate computing other cells and multi-dimensional analysis; H-tree with computed cells can be viewed as stream cube)

Mining Frequent Pattern -- Lossy Counting Algorithm: Major ideas: Not to keep the items with very low support count; Advantage: Guaranteed error bound; Disadvantage: Keeping a large set of traces

Issues in Stream Classification 1) Descriptive model vs. generative model: Generative models assume data follows some distribution while descriptive models make no assumptions; Distribution of stream data is unknown and may evolve, so descriptive model is better **2) Label prediction vs. probability estimation**: Classify test examples into one class or estimate P(y|x) for each y; Probability estimation is better: Stream applications may be stochastic (an example could be assigned to several classes with different probabilities); Probability estimates provide confidence information and could be used in post processing

Stream Classification with Skewd Distribution 1) Problems of typical classification methods on skewed data: Tend to ignore positive examples due to the small number; The cost of misclassifying positive examples is usually huge, e.g., misclassifying credit card fraud as normal **2) Classify data stream with skewed distribution** (i.e., rare events): Employ both biased sampling and ensemble techniques; Reduce classification errors on the minority class

Stream Clustering -- CluStream Framework 1) Tilted time frame work 【Pyramidal Tilted Time Frame: Snapshots of a set of micro-clusters are stored following the pyramidal pattern: They are stored at differing levels of granularity depending on the recency; Snapshots are classified into different orders varying from 1 to log(T) 】**=>** otherwise, will lose dynamic changes **2) Micro-clustering(BIRCH CF-Tree Structure)**:better quality than k-means/k-median: Incremental, online processing, and maintenance **3) Two stages: micro-clustering and macro-clustering**: With limited overhead to achieve high efficiency, scalability, quality of results, and power of evolution/change detection【Divide the clustering process into online and offline components --- **Online component (micro-cluster maintenance):** Periodically store summary statistics about the stream data => Initially, create q micro-clusters (q is usually significantly larger than the number of natural clusters) Online incremental update of micro-clusters (If new point is within max-boundary, insert into the micro-cluster; Otherwise, create a new cluster; May delete obsolete micro-clusters or merge two closest ones **& Offline component (query-based macro-clustering)**: Answers various user questions based on the stored summary statistics; Based on a user-specified time-horizon h and the number of macro-clusters k, compute macro-clusters using the k-means algorithm) 】

11. Socialmedia Mining

Detecting Moving Object Clusters -- Flock and convoy: Both require k consecutive time stamps **Flock:** At least m entities are within a circular region of radius r and move in the same direction **Convoy:** Density-based clustering at each timestamp; no need to be a rigid circle **Swarm:** Moving objects may not be close to each other for all the consecutive time stamps; Efficient pattern mining algorithms for uncovering such swarm patterns

Frequent Movement Pattern vs. Frequent Sequential Pattern: Both aim at finding frequent subsequences from the input sequence database**;** For mining frequent movement patterns, similar places may need to be grouped to collectively form frequent subsequences

Semantics-rich Movement Pattern: In addition to knowing how people move from one region to another, we also want to understand the functions of the regions **A two-step top-down mining approach:** Step 1: Find a set of coarse patterns that reflect people's semantics-level transitions (e.g., office → restaurant, home → gym)**;** Step 2: Split each coarse pattern into several fine-grained ones by grouping similar movement snippets

Pattern discovery in sparse data: Periodicity shows up in some reference "spots" (or "cluster centers")**;** Reference spots can be detected using density-based method**;** Periods are detected for each reference spot using Fourier Transform and auto-correlation

GeoTopic Discovery in Social Media

Previous 1) LDM: Location-driven model; Clustering based on document locations ;One location cluster is a topic **2) TDM**: Text-driven model; Topic modeling with network regularization; Documents that are close in space should have similar topic distributions **3) GeoFolk**: A topic modeling that uses both text and spatial information; The geographical distribution of each topic is Gaussian

Local Event: A local events is an unusual activity bursted within a local area and specific duration while engaging a considerable number of participants

GeoBurst: a reference-based method for local event detection. It consists of three key components: 1) A candidate generator that finds geo-topic clusters in the query time frame, and regard them as candidate events【Intuition: the spot where the event occurs is acting as a pivot that produces relevant tweets around it; Our clustering algorithm is based on: a geo-topic authority score for each tweet; an authority ascent process to find authority maxima as pivots; Computing geo-topic authority (a tweet gets an authority score from neighbor tweets where): Geographical impact: calculated by a kernel function; Semantic impact: calculated by random walk on a key word co-occurrence graph; A pivot is an authority maximum: a prominent tweet that is surrounded by many relevant tweets; A pivot attracts similar tweets to form geo- topic clusters; Find all the pivots in the geo-topic space by Authority Ascent】 2) A ranking module: design the activity timeline structure to summarizes the routine activities in different regions to filter non-event candidates 3) An updater that updates local events in real time as the query window shifts, the strategy is based on the additive property of authority score: subtracting the contributions of outdated tweets; emphasizing the contributions of new tweets.

12. Advanced Classification

CBA Method (Associative Classification): 1) Mine high-confidence, high-support class association rules; 2) LHS: conjunctions of attribute-value pairs); RHS: class labels; 3) Rank rules in descending order of confidence and support 4) Classification: Apply the first rule that matches a test case; o.w. apply the default rule;5) Effectiveness: Often found more accurate than some traditional classification methods, such as C4.5; 6) Why? — Exploring high confident associations among multiple attributes may overcome some constraints introduced by some classifiers that consider only one attribute at a time

CMAR (Classification Based on Multiple Association Rules) : 1) Rule pruning whenever a rule is inserted into the tree **[**Given two rules, R_1 and R_2, if the antecedent of R_1 is more general than that of R_2 and $conf(R_1) \geq conf(R_2)$, then prune R_2**;** Prunes rules for which the rule antecedent and class label are not positively correlated, based on the $\chi 2$ test of statistical significance**]** 2) Classification based on generated/pruned rules **[**If only one rule satisfies tuple X, assign the class label of the rule; If a rule set S satisfies X**:** Divide S into groups according to class labels**;** Use a weighted $\chi 2$ measure to find the strongest group of rules, based on the statistical correlation of rules within a group; Assign X the class label of the strongest group] 3) CMAR improves model construction efficiency and classification accuracy

PatClass(Discriminative Pattern-based Classification) --- Principle: Mining discriminative frequent patterns as high-quality features and then apply any classifier

Framework (PatClass): 1) Feature construction by frequent itemset mining**;** 2) Feature selection (e.g., using Maximal Marginal Relevance (MMR)): Select discriminative features (i.e., that are relevant but minimally similar to the previously selected ones); Remove redundant or closely correlated features 3) Model learning**:** Apply a general classifier, such as SVM or C4.5, to build a classification model

DDPMine (Direct Discriminative Pattern Mining): Efficient, direct discriminative pattern mining **General methodology** 1) Input: A set of training instances D and a set of features F; 2) Iteratively perform feature selectin based on the "sequential coverage" paradigm: Select the feature fi with the highest discriminative power; Remove instances Di from D covered by the selected feature fi **Implementation** 1) Integration of branch-and-bound search with FP-growth mining 2) Iteratively eliminate training instances and progressively shrink the FP-tree

DPClass (Compatible Discriminative Patterns for Linear Models) --- Advantage: can compress the model and thus the online prediction is extremely fast; have comparable performance as advanced models (Even better in experiments); can learn the interpretable patterns (Ex.: Multi-class classification, regression, and survival analysis) **Implementation:** 1) Generation of discriminative patterns**:** Based on Random Forest; 2) Select a k-set of most compatible discriminative patterns (Forward Selection: Greedy; Lasso: GLMNET)

Complementary Questions

How to design a stream data cube to facilitate watching power consumption fluctuation in Illinois and be able to drill down by time, location, etc.? For the stream data cube of power consumption, we first adopt the tilted time frame strategy to model different time granularities, such as second, minute, quarter, hour, day and week. We define two different critical layers on the stream data cube: minimum interest layer (m-layer) and observation layer (o-layer), in order to observe power consumption fluctuations at o-layer and occasionally drill-down down to m-layer. We partially materialize a subset of cuboids along the popular drilling paths and make use of H-tree structure for efficient storage and computation.

Explain why lossy counting algorithm can find the frequent items with a guarantee error bound . Given the length of the data stream seen so far as N, Lossy Counting (LC for short) divide the stream into buckets of size. At each bucket boundary, LC decreases the count of items by 1. In this way, the frequency count error is less than the number of buckets, which equals . And true frequencies of false positives are lower-bounded by . Note LC does not generate false negatives.

Classifier for highly screwed data in dynamic evolving data streams. As the data is highly skewed, we make use sampling based methods to compose the training data set and incorporate positive examples to increase the training set size. In such a way, the variance of data bias can be effectively reduced. We further make use of an ensemble method for classification in order to reduce variance caused by a single model. This sampling and ensemble based classification techniques can be effectively used for classifying highly skewed data in dynamic data streams.

Classification: descriptive model (no assumption) is better than generative models; Probability estimation is better than label prediction

VFDT: Based on Hoeffding Bound principle, classifying different samples leads to the same model with high probability —can use a small set of samples. Scales better than traditional methods; Incremental; Could spend a lot of time with ties; Memory used with tree expansion; Number of candidate attributes

CVFDT: Increments count with new example, Decrement old example (Sliding window), Nodes assigned monotonically increasing IDs, Grows alternate subtrees,

When alternate more accurate, replace old, O(w) faster than VFDT-window

Ensemble: train K classifiers from K chunks, for each subsequent chunk: train a new classifier, test other classifiers against the chunk, assign weight to each classifier, select top K classifiers

Concept drifts: Define and analyze concept drifts in data streams. Four possibilities: No change; Feature change: only P(x) changes; Conditional change: only P(y|x) changes; Dual change: both P(y|x) and P(x) changes. The expected error could increase, decrease, or remain unchanged. Training on the most recent data could help reduce expected error.

Among three popular classification methods: (1) Naive Bayesian, (2) support vector machine, and (3) k-nearest neighbor, which one is easy to be adapted to the classification of dynamically changing data streams, and how? and which one is difficult to do so and why? kNN is easy to be adapted in evolving data streams, because it is a lazy learning method and does not need a model based on previous data, so evolving data stream would not affect the method. Both Naive Bayesian and SVM are difficult to be implemented in data stream, because these methods require training process. With the dynamical changing in the stream, we need to detect concept drifting and re-train the model over and over again. The training process of SVM is much more difficult than Naive Bayesian, so SVM is the most difficult method to be implemented in evolving data stream classification.

Explain why LTM (Latent Truth Model) is more powerful than TruthFinder on truth finding.! [3 Points] TruthFinder: Quality as a single value, i.e., Precision or Accuracy [4 Points] LTM: two major differences: (i) handling multi-values truth (e.g., a book may have multiple authors, i.e., multiple truth; Note: TruthFinder handles only single-valued truth, if an information provider gives only 2 names for a 3-authored book, it is simple wrong); [4 Points] (ii) Two-sided errors, i.e., false positives and false negatives (i.e., if A gives two names but miss the 3rd author (false negative), B gives two right names but one wrong name (false positive), C gives three right names; D gives 3 right ones but one wrong name, you can easily distinguish them using LTM but not in TruthFinder).

We can use the following two NLP methods to help find phrases.

• **Chunking**: chunking models are able to identify the constituents in a sentence. As most phrases are constituents in practice, we can use chunking models to help compute the scores of candidate phrases, which may give better phrases.

• **POS Tagging**: pos tagging models aim at assigning parts of speech tag to each word. As most phrases consists of several noun words, we may use the pos tags to help find high quality phrases from the candidate list.

Not every piece of news or tweets is truthworthy. Design a mechanism that may use both sources to identify what is likely to be the truth in news and tweets

Our basic intuition is that those truth facts are both identified in news and tweets will have a high confidence to be true and the same as those false ones. The basic mechanism is to do co-training on both news and tweets. That is for those we identify as truth facts, we check them in the news and vice verse. We could do it iteratively and finally come to a stable stage.

You are required to construct a typed heterogeneous CS information networks from such a database. The network should contain not only bibliographic information (author, venue, title, year) but also detailed research theme information for each publication, such as "deep learning", "frequent pattern mining", etc. Outline your design of a set of methods that may construct such a network effectively. (15 points)

The main task is to construct the bipartite graph between papers and key-words. To do that, we can first use SegPhrase to generate key phrases and use them as the theme nodes of the network. After that, we need to infer the key phrases of each paper. Here we can use Latent Keyphrase Inference(LAKI), which is powered by Bayesian Inference on Quality Phrase Silhouette. We could get document representations using key phrases. Finally, we could build the network effectively using the above set of methods.

Other Version

First, Phrase Mining -- Phrases -- Entity Typing (filter out research theme)

Second, Link the theme with publications(paper), mine phrase from each paper and link them with relative theme