Summary for Mini Turbo Syncer

Xiaoyao Qian

Abstract

Turbo Syncer[1] is the method proposed to segment fields in unstructured text. It utilized the peer sources of information from different domains(by domain I mean personal websites) to extract and cross correct the information fields accordingly. I proposed a Mini Turbo Syncer that will handle the cases where the format and order are not guaranteed to be consistent inside of one domain. Mini Turbo Syncer(MTS) is designed to cross correct the information extraction in one particular domain.

1 Overview

- The model used is Hidden Markov Model.
- The publication records will be first tokenized into tokens, and then translated to binary feature vectors through some feature functions.
- Segmenting publication record into different fields is done by applying Viterbi Algorithm, which will label each feature vector with a label(TI-Title, FN-Firstname, LN-Lastname, VN-Venue, YR-Year, DL-Delimiter)
- Constraints will be applied to the resulting labels. For example, TI labels have to be continuous; if there is a VN in between TI labels, then it is not allowed.

2 Pipeline

The pipeline of the MTS for ONE particular domain(personal publication page) can be broadly divided into 6 steps:

- Data Preprocess
- Initial Segmentation

- Background Knowledge Model(BKM) Building
- Boosting Segmentation
- Combine Stage
- Constraint Validation

Details steps and procedures of the pipeline are described below

2.1 Data Preprocess

Data Preprocess involves several steps. First the system will retrieve and generate training samples from structured sources of publications. e.g Google Scholar, Microsoft Academic Search, ResearchGate, etc. The records from these sources are structured so that it is easy to get the author fields, title fields, venue fields and date fields for a particular publication record.

Regenerate different versions of one piece of record in the training sample by changing the structure order. For example:

Beyond Pages: Supporting Efficient, Scalable Entity Search. [Title]
T. Cheng and K. C.-C. Chang. [Author]
In EDBT, pages 15-26, [Venue]
2010.[Year]

can be regenerated into different versions by re-arranging the order of Title and Authors:

T. Cheng and K. C.-C. Chang. [Author]
Beyond Pages: Supporting Efficient, Scalable Entity Search. [Title]
In EDBT, pages 15-26, [Venue]
2010.[Year]

There are many ways of regenerating different versions of training samples (e.g. Changing the order of firstname and lastname; changing delimiting notation from comma to period). The purpose of this step is to expand the size of training sample, so that during the training step, the model to train will see more patterns and thus enrich its knowledge when calculating initial probability, transition probability and emission probability(the three parameters in an HMM model). I adopted this methodology from the "Single Image Recognition" problem from the area of Computer Vision. One

heuristic way to resolve this CV problem was to generate virtual images by processing the origin single image(by distorting, etc.) and so the training samples get larger[2]. At last, the raw publication records and their variations will be tokenized and translated to feature vectors.

2.2 Initial Segmentation

Supervised train the HMM model M1 with the training samples and also variations of the training samples. Segment the publications in this domain. Denote the segmentation results as S1.

2.3 Background Knowledge Model Building

Construct a Background Knowledge Model based on S1. The model I chose was a simple unigram language model that record the label distributions for every feature vector. The BKM will be useful in the later combining stage.

The intuition for this BKM is to record the label distribution for feature vectors IN THIS DOMAIN.

2.4 Boosting Segmentation

Supervised train a new model M2 with S1 as training samples. Segment the publications in this domain again. Denote the segmentation results as S2.

2.5 Combine Stage

Combine the segmentation results S1 and S2. Now that for each feature vector(noted that the representation of a piece of publication is a sequence of feature vectors) in the publication has two labels. One from S1, and the other from S2. The combiner will decide which of the two labels is more likely and more reasonable to be placed here. This decision process involves looking up the label distribution in the BKM constructed earlier, and checking constraints of labels at the given position. After the combining step, we should have a final segmentation, denoted as S3.

2.6 Constraint Validation

Final step is to validate S3 against constraints again. Make sure no label is against the constraints. If there are labels that are against constraints, it will find the second most probable label there and do the validation again until all the labels are fine.

3 Assumptions & Constraints

The basis that support the MTS to work ideally is the assumption that the majority of records are labeled correctly in the Initial Segmentation. This correctness will provide a good set of retraining samples for the Boosting Segmentation, and also will provide a confident BKM for Combine Stage. If the results from the Initial Segmentation have a lot of mislabels, then the boosting stage will make the segmentation results even worse, as well as the Combine Stage.

The constraints applied to the model involves:

- Continuous Label Constraint: Labels sequence should be continuous. For example, Title labels should be continuous, namely there should not have any label other than Title inside of a Title label sequence.
- Label Order Constraint: Possible order of label sequences can only be: Author→Title→Venue or Title→Author→Venue. This is based on the observations on the current data in the Grantforward Database. This is a domain-specific constraint.
- Token Level Constraint: If some token is seen, it must be related to a certain label. For example: all commas should be labeled as delimiter; all 4-digit number ranges from 1980 to 2013 should be labeled as year; no words should be considered as Delimiter, only punctuation could be possible Delimiter.
- Delimiting Constraint: Some label sequences must be separated by a Delimiter label. For example: Titles cannot be stop immediately and turn into Venue labels; there must be a Delimiter label in between Title sequences and Venue sequences.

These constraints are helpful because it will help remove all the impossible cases away when doing the Viterbi decoding during labeling.

4 Modified Viterbi Algorithm

Here I want to talk more about the modified Viterbi algorithm I used in the model to do the labeling work. Naive Viterbi Algorithm related to the HMM is pretty well developed. The Viterbi algorithm is using a dynamic programming way to calculate that: for a given observation sequence, what label sequence(state sequence) in the whole search space is most likely to be corresponding to the observation sequence. However, since the naive Viterbi algorithm always returns the label sequence that have the highest probability to be correspond to the observation, the resulting label sequence might contain invalid transitions according to the constraints. So the modified Viterbi will add one more step to the naive Viterbi, which is to verify constraints along with its dynamic programming process. If at a certain step, the Viterbi decided on a label, but we found it against the constraints, then the Viterbi will set this label(and also the precedent labels) a low probability, so this "path" will not be likely to be referred to again in the following decoding steps. The most important insight to bear in mind for Viterbi algorithm is that: if a label is most suitable for a observation at this step, it doesn't mean that this label will be the best fit. One example is:

Molnar, T.J., E. Water, J.M. Capick...

When the model is labeling the above sequence(a sequence of firstnames and lastnames), it is labeling following the order of sequence, so the first step will be:

1. Molnar: very likely to be Lastname.
2. Comma: Delimiter
3. T.J.: likely to be abbrev. of Firstname
4. Comma: Delimiter
5. E: likely to be abbrev. of Firstname
6. period: Could be a delimiter, indicating the end of the name fields; but also can be abbrev of firstname.
7. Water: An ENGLISH WORD! Likely to be a start of title.

Note that if the sequence stop at here, the "Water" will be labeled as Title, because if we are only watching the part "Molnar, T.J., E. Water", the "Water" is more like a Title. P("Water"|Title) is thus the largest among all conditional probability of "Water".

But if we move on, the later tokens will come in and lower the probability of "Water" being Title(because the following patterns are still in name

pattern, so it will lower the probability of "Water" being Title, and choose to decide that "Water" being Name).

5 Experiments

6 sets of experiments are done on 6 different domains(Zhai, Hahn, Kevin, Yoon, Rhinesmith, Sathuvalli). Among the 6 domains, 4 of them have more than 70 publication records; the other 2 have less than 20 publications. The results are shown as followed:

Domain	#Pub	#MissIn1st	#MissIn2nd	#MissInCombine
Zhai	126	6	36	10
Hahn	92	15	9	6
Kevin	87	8	12	8
Yoon	71	11	8	5
Rhinesmith	13	4	5	3
Sathuvalli	13	3	5	3

Later I did the second analysis and experiment, after the feedback from Kevin. And during the second experiment, I did some change in both training process, and feature construction, the changes I made will be briefly stated below, and the reason for such changes will be explained in section 6.

1) Change the training process: Originally, the training process in the second iteration uses same methodology as the first iteration: train with smoothing. Smoothing here means applying Laplace smoothing to the observation patterns that is not seen in the training set. Namely, for every transition and observation not seen in the training set, the P(Transition not seen in training set|Prev State) will be 1/number of all possible transition; P(Observation not seen in training set|State) will be 1/number of all observation, instead of both 0. Now in the second iteration, the training process will not apply smoothing any more. The reasons will be stated in section 6.

2) Added more distinguishable feature in both first and second iterations. Details will be explained in section 6.

2nd Experiment Results

Domain	#Pub	#MissIn1st	#MissIn2nd	#MissInCombine
Zhai	126	7	5	7
Hahn	92	15	9	6
Kevin	87	8	7	7
Yoon	71	11	8	5
Rhinesmith	13	4	3	4
Sathuvalli	13	2	2	2

The 2nd column is the **number of publication in this domain**; 3rd column is **how many record has mislabel in its first iteration segmentation**; 4th column is **how many record has mislabel in its second iteration segmentation**; 5th column is **how many record has mislabel after combine stage**.

6 Problems & Analysis

I've done a more thorough analysis on the experiment results after hearing the feedback from Kevin. And now all mislabeling can be categorized into the following problem sets.

6.1 Ambiguous Tokens

One of the errors appeared in the 2nd iteration segmentation is mislabeling on some ambiguous token. Let's see an example:

Kavita Ganesan, ChengXiang Zhai, Jiawei Han. Opinosis: A Graph Based Approach to Abstractive Summarization of Highly Redundant Opinions, Proceedings of COLING 2010, pages 340-348.

This is one piece of record from Zhai's page, where most of the names on Zhai's page are non-abbreviated; namely, the author names appeared in Zhai's publication are all written in full format(as you can see from the above record). The mislabel during 2nd iteration in this record happened on "Opinosis". This word is not an English word; it is capitalized, and its previous token is "Han" and a period, which are highly possible to be names. So here, the word "Opinosis" is very hard to tell which label it it meant to be. It can be a Title(which it should be), or it can also be a name following the previous name patterns. I monitored the feature vector for the token "Opinosis", it is [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0], this feature vector is exactly same as the feature vector for to-

ken "ChengXiang"(which obviously should be a Firstname). And the label distribution from the **1st iteration result** for this feature vector pattern is:

Label	Count	Frequency
FirstName	147	96.08%
Title	5	3.27%
Venue	1	0.65%
Lastname	0	0%
Delimiter	0	0%
Year	0	0%

This label distribution tell us that: For this feature vector, mostly(96% of the time) the model labeled it as Firstname **in the other parallel records in this domain** during 1st iteration, and very rarely it was labeled as Title. Therefore, during the 2nd iteration training process, where we use the results from the 1st iteration as training sample, this feature vector pattern is seen more as a Firstname rather than Title. And so the emission probability P(this feature vector pattern | Firstname) >> P(this feature vector pattern | Title). Due to this change in emission probability, during the labeling process, the newly trained model will recognize this feature vector pattern as Firstname, simply because when it is labeling, it uses Viterbi Algorithm I described earlier, and Viterbi Algorithm will decide by finding the label with highest probability. Here Firstname will contribute a higher probability than Title when Viterbi is calculating, so this is why the word "Opinosis" is labeled as Firstname.

This kind of problem takes up 90% of all errors. After some speculations on the patterns and mislabels, I discovered that one of the common mislabeling happened to publication title pattern like:

NewNotion: Explanation to the new notion proposed...

The NewNotion are often some capitalized, non-english words or phrases(e.g. Opinosis, RoadRunner, Micropinion Generation, PowerBookmarks, DirichletRank, etc.) and can be easily mislabeled to Name field. This is because that for this type of title pattern, the existing features is not capable to tell the difference. So we need more distinguishable features to distinguish these patterns. Therefore, I added a new feature corresponding to such patterns(including: Newnotion, NewNotion, NewNewNotion patterns followed by a colon) to the feature set. Now for such NewNotion title pattern, the

feature representation can be more distinguishable between Name and Title. And the additional feature really improve the accuracy(see 2nd iteration miss rate improvement between first experiment and second experiment on Kevin and Zhai's page where such cases are common)

Another example:

Do, H., Telionis, D.P., and Yoon, R.H., Foam Drainage and Wetting: a Numerical Investigation, Proceedings of the 8th International Congress Fluid Dynamics and Propulsion, Sinai, Egypt, ICFDP8-EG-170, 6p. (2006).

The first token "Do" is very ambiguous here. It is very likely to be a Title, and since it's at first position of the publication, it can be both possible to be Title or Name. During 1st iteration segmentation, this token is very likely to be mislabeled to Title(but actually it's a Firstname). However, luckily "Do" also appeared as author in the parallel records in this domain, and in those parallel records, this "Do" is not necessarily the first author(namely "Do" is placed at some following position), and in those cases, "Do" are labeled correctly as Firstname(because of constraints). So in the 2nd iteration, the "Do" appeared at the first position can be corrected.

To address the ambiguous token problem, I noticed that if for the same token, majority of this token is correctly labeled, then this token should be cross-corrected. To be able to distinguish such token, I think of the same idea as in the case of NewNotion. We need more distinguishable features in the second iteration. I added a new feature in the feature set in second iteration: if the token appeared repeatedly and is labeled as name field in the background model(from first iteration). The background model is based on the segmentation results from the first iteration, which will represent the majority labeling on a specific token. These two features did solve some of the ambiguous token problems, but it also brings side-effect: It is hard to decide how frequent a token appears can be categorized as "appeared repeatedly". This parameter is currently setup empirically.

There are still remaining problems that can be categorized into this kind of problems. And to think of new features for every ambiguous problems is non-trivial. For example:

Hahn, J. (2013). The Best 100 Free Apps for Libraries. New York: Rowman & Littlefield Publishing Group.

In the above record, "The Best 100 Free Apps for Libraries" are labeled as Venues. When I look into their feature representations, 60% of the labels related to these feature patterns were Titles, and the remaining 40% of the labels were Venues. There isn't a dominant label here. If there could be more distinguishable features to represent the tokens in the second iteration, the titles and venues can be easier to distinguish.

But if we cannot think of more distinguishable features, there is one last proposal which is to train the HMM models in a tri-gram way, rather than the bi-gram way that I'm currently using. Normally, in POS and NER-like tasks, using a tri-gram will more or less improve the bi-gram results. However, speed is also one thing we should take into consideration. Currently, performing 6 individual segmentation tasks on 6 domains from training models to combine steps will take up 45 min. Training tri-gram models must will increase the running time.

6.2 Weak Delimiter Recognition

Weakness in recognize delimiter is also one of the reasons that causes the mislabeling. This situation mostly happens to recognizing "." as delimiters. Because period can appear in author name field as abbreviation(e.g. K.Chang). It can appear in title fields as abbreviations as well(e.g. A vs. B). It can appear in venue fields as abbreviation as well(e.g. XXX Conference, St.Louis). It can also be delimiter between fields. Since it has so many different usages(way more than commas), it takes more effort in labeling periods, and sometimes mislabel periods can break the constraint balance. For example:

Sathuvalli, V.R. and S.A. Mehlenbacher. 2012. De novo sequencing of hazelnut bacterial artificial chromosomes (BACs) and targeted marker development for disease resistance using multiplex Illumina sequencing. Tree Genetics and Genomes

The part "Tree Genetics and Genomes" is the name of a journal, and since from the feature perspective, they are nothing big different from titles(they are all english words, and capitalized). If the period before "Tree" is correctly labeled as Delimiter, then according to the Continuous Label Constraint, the "Tree Genetics and Genomes" will no longer be able to be labeled as Title, and so only left with Venue.

I applied a similar methodology that is to think of additional feature here to make the delimiters more distinguishable. One additional feature is that if the token appeared repeatedly and is labeled as delimiter in the background model. This solves some of the mislabeling, but for the above example, the period is used mostly as abbreviation in Firstname, so the new feature will tell that this period is mostly Name field, and still cannot distinguish the period between title and venue as delimiter. For such cases, I think it would be effective to hardcode some rules addressing periods in the combining step. Namely, if Name fields has been passed, among the remaining tokens, periods can only be delimiter.

6.3 Side Effect of Continuous Label Constraint

One of the typical mislabeling error is due to the side effect of delimiter constraint. For example, if there is a piece of publication record that has commas in titles like the following:

PowerBookmarks: A System for Personalizable Web Information Organization, Sharing, and Management. W.-S. Li, K. C.-C. Chang, D. Agrawal, and et al. In Proceedings of the 1999 ACM SIGMOD Conference (SIGMOD 1999), pages 565-567, Philadelphia, Pa., June 1999

The model will label title field as: "PowerBookMarks: A System for Personalizable Web Information Organization". Because the following token is the "," and will be labeled as Delimiter. According to the Continuous Label Constraint, Title label sequence must be continuous, and could not be split by any other labels.

This kind of problems is non-trivial. I applied a draft way to address this problem, but the approach does not cover much situations: I spotted that if a comma appear and is followed by a first-letter-lowercase token, this comma is very likely to be inside of title. So I strip such commas in the tokenizing step. However, it is not always the case. Some commas with following first-letter-capitalized token could also be inside of title. And such comma pattern could also appear in Name fields or delimiters that separate different fields. So I cannot strip the commas so easily.

6.4 Invalid Format & Complicated Format

Since the publications is crawled from the internet by a general-purpose crawler, sometimes some record that are not actually publication will be stored into our database. These records are of invalid format, and may not be correctly labeled. This problem is non-trivial, and the reason for such problems are caused by the ill-formatted publication page the user submitted. Since we've made quite a lot of optimization on the publication extraction, if the publication page is in good format, there should be no problem with extracting valid formatted publications to segment, but if the publication page is ill-formatted, there's really nothing to do in the segmentation process.
Some publications may have complicated format, where the constraints we set may be too strict for them, and thus will cause mislabel. For example:

Wolske, M., Rhinesmith, C., & Kumar, B. (under review). Community Informatics Studio: Designing Experiential Learning to Support Teaching, Research, and Practice. Journal of Education in Library and Information Science.

This is a piece of record with complicated format. The "under review" can be labeled as Venue, but later the "Journal of Education in Library and Information Science" are also Venue, and it is against the Continuous Label Constraint.

Zhang, J. and Yoon, R.H., Hydrophobic Forces Mewasured between Silica Surfaces Coated with C TACl Homologues, in Interfacial Phenomena in Fine Particle Technology, Z. Xu and Q., Liu., Editors, Proceedings of the Sixth UBC-McGill-UA International Symposium on Fundamentals of Mineral Processing, Oct. 1-4, pp. 45-60 (2006).

This is another piece of record with complicated format, where it seems to have two different sequence of name fields.

Xuanhui Wang, Jian-Tao Sun, Zheng Chen, ChengXiang Zhai,Latent Semantic Analysis for Multiple-Type Interrelated Data Objects Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'06), pages 236-243.

This is an ill-formatted publication record, where there's no obvious

delimiter between titles and venues. For ill-formatted records, I have a proposal that is to build a boundary token model based on the segmentation results from the first iteration. This boundary model will find out the most frequent tokens at the transitioning spot(when Title turns to Venue, When Names turns to Title, etc.). So that with this boundary model, if there's a token in the ill-formatted record is a frequent boundary, then it is very likely there is a boundary there which was omitted by the previous iterations. The tokens in Boundary model will have to keep their raw format. For example, some researchers will use "In xxxx" as the start of venues. Here, "In" is the boundary of Venue, we'd like to keep the "In" capitalized, because "in" might be less distinguishable than "In".

6.5 Others

Apart from the above problems, there were some amount of mislabeling that really doesn't make sense, and cannot be categorized into any of the above problem sets. For example

Wolske, M., Rhinesmith, C., Archer, J., Bayci, E., Leuzinger, R., McKeever, L. (2013, February). "Community Informatics Studio: A Model of Information Scholarship in Action." Poster presented to the iConference 2013, Fort Worth, Texas, USA.

The "Rhinesmith" is mislabeled to Venue, although this token is the most popular token throughout the whole domain(it's his page), but from the feature level, the representation of the token(preceding by a delimiter, and followed by a delimiter), is not very common in this domain(mostly Rhinesmith is the first author). But anyway from its label distribution, the dominating label is still Name field. Why is this mislabel happen? Later I discovered that such situation happened in small number sample domains. And the reason for this mislabeling is because I used the smoothing method in the second iteration training process. The smoothing is not required in the second iteration training, because the training set and test set in second iteration are the SAME records. Thus, even if there exist some transitions or observation pattern that is not seen in the training set(which will be the segmentation result from the first iteration), those observation will not be evaluated during the second iteration anyway. However, if applying the smoothing, the probability of unseen transition and observation will be raised(although raised by small amount, but this amount could be disastrous in small number sample domains), and thus during the calculation

of Viterbi, those unseen transitions and observation will be treated equally possible as the others. This is very important especially for those domain with small number of records. Let's take the "Rhinesmith" case as example: assume the pattern for that particular "Rhinesmith" is denoted as A, and the fact that pattern A is labeled as Lastname only appeared once in the training set of second iteration:

P(A | Lastname) = 1/M

Now if applying the smoothing during the training process, some other emissions, although have never seen in the training process, could also be possible by the Laplace Smoothing:

P(A | Title) = 1/M

P(A | Venue) = 1/M

...

And so when the Viterbi algorithm is doing the labeling work, it will treat this A equally likely to be labeled as Lastname, Title and Venue. Since the purpose of smoothing at the first iteration was to prevent unseen transitions and observations to return 0 in probability calculations. Such situations would never happen in second iteration, so it is perfectly fine to not apply smoothing during second iteration training. After removing the smoothing process, remaining problems can always be categorized into one of the above categories.

7 Future Work

After the summary, I found there was space for improvement in context awareness in between parallel records in the same domain. One of the thought is to further improve and extend the knowledge extent of the Background Knowledge Model. Particularly, the tokens and feature vector pattern around label changes boundaries. For example, when title sequences stops and venue sequences starts, there should be some common pattern at this boundary point. If we can find these patterns, it can be applied in the combining stage.

8 Potential Application in GrantForward

Although the segmentation can never be 100% perfect, one way I could think of to integrate the publication segmentation to the GrantForward Profile system is that: we can set up a Solr search engine in the backend, and dump all the segments into it. And so if a user wants to click on one publication and expect to see who else established in the same venue, we could sent the venue field segment of this clicked record as a query into Solr, and then ask Solr to search in all of the venue segments we have, and find the most similar results to return. In this way, even though the venue may contain some noise words, Solr should be able to handle that when it tries finding the most similar results.