АВТОМАТИЧЕСКОЕ ПОПОЛНЕНИЕ БАЗЫ ИМЕНОВАННЫХ СУЩНОСТЕЙ НА ОСНОВЕ ПОЛЬЗОВАТЕЛЬСКИХ ЗАПРОСОВ

Кудинов М. (m.kudinov@samsung.com),
Пионтковская И. (p.irina@samsung.com)

Исследовательский Центр Самсунг, Москва, Россия

В работе описан процесс пополнения базы именованных сущностей, предназначенной для поддержки голосовых команд на устройстве. Пополнение производится полностью автоматически без участия редактора путем анализа логов запросов к диалоговой системе. Логи состояли из откликов системы распознавания речи и поэтому содержали большое количество ошибочных распознаваний именованных сущностей. Поиск подобных ошибок также осуществлялся в рамках описываемого подхода. Используются методы на основе взаимной информации и скрытых марковских моделей.

Ключевые слова: скрытые марковские модели, редакторское расстояние, система распознавания речи, именованные сущности, анализ поисковых запросов

AUTOMATIC UPDATE OF THE NAMED ENTITIES DATABASE BASED ON THE USERS QUERIES

Kudinov M. (m.kudinov@samsung.com),
Piontkovskaya I. (p.irina@samsung.com)

Samsung R&D Institute, Moscow, Russia

We describe an algorithm of update of the database of named entities providing support of voice commands on a device. The update is made automatically with no human assistance by means of analysis of query logs of the dialogue system. The logs consisted of responses of the automatic speech recognition engine and thus contained erroneous recognitions. The search of such mistakes is also made as a part of our method. The problem of named entities extraction was solved by means of an algorithm based on entropy and mutual information statistics. The detection of recognition mistakes was made by means of a novel data-driven probabilistic approach taking into

account grapheme substitution statistics in the data. Assuming grapheme alignment hidden, we use the EM algorithm for training the model. As a result we obtain a statistical model capable for sequence similarity assessment. The algorithm based on our similarity score performs better in terms of F_1-measure than one using the classical Levenshtein distance.

Keywords: Hidden Markov models, edit distance, speech recognition, named entities, search query analysis

1. Problem statement. Dialog system

Consider the following simple scheme of the voice commands engine working as follows:

1) The user's query is recorded using the embedded microphone. The recorded speech signal is input to the speech recognition system which outputs text string sometimes containing errors.
2) The text is input to the query grammar based parser.
3) The parser outputs the semantic frame which determines the system response.

The system needs named entities database for parsing of queries like

покажи мне мультфильм пингвиненок пороро / show me the cartoon pororo the little penguin

я хочу посмотреть новый сезон сериала ментовские войны / I want to watch the new season of the series cops wars

The problem of the database support may be solved by means of collecting a gazetteer grabbing named entities from program guide or film ads but this approach has two complementary drawbacks: a) such list would have contained shows no one ever watches; b) the list would have missed popular video clips on YouTube, Vimeo etc.

The other problem caused by gazetteer based approach and an external speech recognition engine (ASR) is the need of interpreting queries containing ASR errors. A trivial analysis has shown that for a series Magnificent Age popular among Russian housewives the proportion of queries where the film name undergoes different changes reaches 5%. Adding these variants to the gazetteer is a necessary measure for providing good performance of the service.

Thus, we have to solve two tasks:

1) Find film names (TV programs, actors and directors names etc.) absent in the gazetteer and spelled right;
2) Find different variants of recognition of the objects in the gazetteer and add them with corresponding mark.

The problems will be considered below in corresponding sections.

2. Named entities extraction

The problem of the multi-word named entities extraction in short queries may be easily converted into the problem of collocation extraction. The classical approach to this task is the calculation of the pointwise mutual information between tokens:

$$PMI(x;y) = \log \frac{f(x,y)}{f(x)f(y)},$$

where $f(t)$ is the frequency of the token sequence t in corpus. PMI measures mutual dependence between two random events (x,y). All the token pairs found in the corpus were included into the list sorted by calculated PMI. It was the way the top list for the collocations of length 2 was made up. For the search of sequences of three tokens we take minimum of two calculated PMI scores:

$$PMI(x;y;z) = \min\big(PMI(x;yz), PMI(xy;z)\big).$$

The PMI for longer sequences was calculated the same way. We made experiments with the sequences of the length up to 6 (*Чип и Дейл спешат на помощь/ Chip ‹n Dale Rescue Rangers*). The top list on PMI for the sequences of length from 2 to 4 is shown in the column 1 of the table 1. It is obvious that some sequences are prefixes and suffixes of others.

The method of sequence merging was based on the notion that the $PMI(w_1 \dots w_t)$ of the named entity sequence is: a) greater than the $PMI(w_1 \dots w_{t-1})$ of its prefix; b) greater than the $PMI(w_1 \dots w_{t+1})$ of the sequence $w_1 \dots w_{t+1}$ with the prefix $w_1 \dots w_t$.

The experiment has shown that such notion leads to good results.

We used the following algorithm for n-gram merge. Firstly, the n-gram prefix tree was constructed such that every partial way corresponded to a n-gram and each node contained corresponding calculated PMI.Further, the tree was traversed such that each time the PMI of longer path was less than the PMI previous partial path the algorithm output the partial way and switched to the next branch. The sequences having a common suffix were merged according to the same principle.

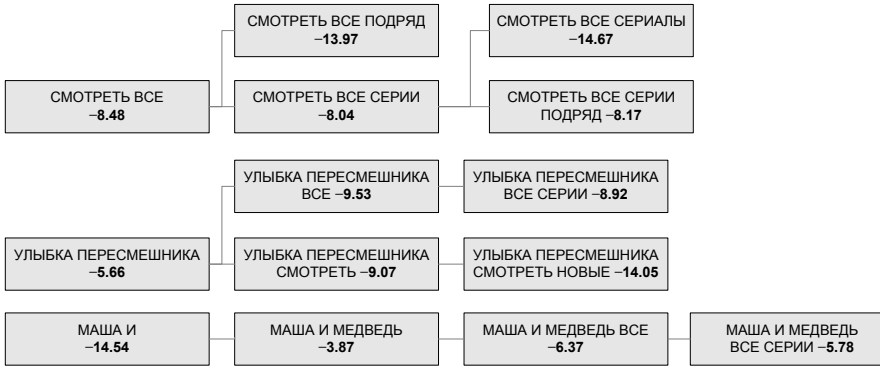

Fig. 1. Prefix tree for n-gram merge

We also did the same operation in the reverse order with the suffix tree. We had to do it because the forward pass missed the sequences with high entropy of the prefix: **Three** days to kill.

The result of the described algorithm is shown in the column 2 of the table 1.

The last step we need to extract named entities is separate true named entities from other frequent collocations. There are at least two approaches to this problem. The first one is to take the hypothesis that named entities have similar distributions of the neighboring words in the query. For example, the cartoons and TV shows are often found with the words *watch, season, episode* etc. It would make possible using of any reasonable linear classifier e.g. logistic regression or SVM. But we revealed that the named entities may be detected reasonably well only according on the entropy of this distribution. As far as entropy characterizes the degree of "uniformness" of the distribution, the distributions with high peaks should have lower entropies. It is obvious that the word *film* which potentially can be found in context of every named entity with equal probability, should have greater entropy than a series name. After the final sort on entropy the top list looked like in the column 3 of the table 1.

It is easily seen that the top list is rather homogenous sample of film names, actors names and pop stars. At the same time we see that the most popular films did not get to the top list because they had too many different contexts. This result has an important advantage because it detects named entities which have little chance to get to a manually collected gazetteer. The other remarkable issue here is the top-1 entry which is not named entity. This strange result is caused by abnormal frequency of the n-gram *чужие два фантастика боевик*: this whole request was encountered 15 times of all 20 appearances of the film *Aliens* in the search log.

Table 1. N-gram top list

Sort by PMI	**After n-gram merge**	**Sort by Entropy**
маша и медведь	маша и медведь	чужие два фантастика боевик
маша и	http //www google com	серая шейка
и медведь	www yandex ru	спортлото восемьдесят два
http //www google com	три d	самолеты огонь и вода
www yandex ru	две тысячи четырнадцать	руби и йо йо
* *	* *	феи загадка пиратского острова
три d	улыбка пересмешника	секс по дружбе
две тысячи четырнадцать	vk com	мистер и миссис смит
http //www google	новые серии	фильмы для взрослых
//www google com	все серии	teenie лав
yandex ru	физрук второй сезон	огги и тараканы
www yandex	www яндекс ru главная	полли робокар

Sort by PMI	After n-gram merge	Sort by Entropy
vk com	экстрасенсов пятнадцатый сезон	мультфильм тини лав
улыбка пересмешника	смотреть онлайн	крошка енот
все серии	битва экстрасенсов	отпуск по обмену
смотреть онлайн	google com	осторожно обезьянки
второй сезон	чернобыль зона отчуждения	бен и холли
физрук второй сезон	сын за отца	тотальная распродажа
тысячи четырнадцать	д * *	блондинка в эфире
новые серии	ха ха ха ха ха	идентификация борна
все серии подряд	ну погоди	дьявол носит prada
www яндекс ru главная	свинка пеппа	люди икс дни минувшего будущего
http //www	серии подряд	федорино горе
google com	человек паук	большое зло и мелкие пакости
//www google	ж * * у	новогодние приключения маши и вити
б * *	в ж * * у	пингвинёнок пороро
битва экстрасенсов	финес и ферб	паровозик томас и друзья
* * а	черепашки ниндзя	по дороге с облаками
д * *	физрук два сезон	котёнок по имени гав
маша и медведь	маша и медведь	притворись моей женой

3. ASR system errors detection

Let us now turn to the second problem. Assume that we managed to extract new good named entities and we have log of the ASR engine. Assume now that we can find word sequences according to the mask *I want to watch the film <FILM NAME>*. We believe that the placeholder FILM NAME is filled with an actual name of some movie. Now if the FILM NAME is not present in the gazetteer and its frequency is low we consider it as a candidate to the list of wrong recognitions of named entities and we must match it against gazetteer entries.

The last problem is almost classical statement of the sequence alignment problem which often emerges in the natural language processing [6] and speech recognition [4]. Speech recognition is also connected with grapheme-to-phoneme conversion ([2], [3], [1]), another important sequence alignment problem. [1] describes the approach based on so called graphones i.e. letter-phoneme pairs with and Baum-Welsh-like learning algorithm.

This approach leads to the method of similarity distance measure alternative to the popular Levensthein distance. Levenshtein distance gives equal penalty $d(x,y)$ to any deletions, insertions or replacements in the string S_1 relative to S_2. But the errors made by ASR systems tend to emerge because of replacement of a word w_1 with similarly

sounding word w_2. In that sense it would be reasonable to use a metric where $d(x,y)$ is higher for those pairs of letters x,y, which are more likely to be in replacement pairs ("o"–"a", "м"–"н" etc.), and lower otherwise. It is also logical to take the data driven approach where $d(x,y)$ is calculated from data. Then we can take the probability of replacement of the letter x with the letter y in the recognized string as a measure $d(x,y)$. We will test the performance of the Levenshtein measure with the one proposed here.

Consider the probability that the string S will be recognized as a string R:

$$\mathbf{P}(S,R) = \sum_{\{L_{S:R}\}} \mathbf{P}(S,R,L)$$

where $\{L_{S:R}\}$ is a set of all alignments between S and R.

Let $L_{S:R} = l_1, l_2, \dots l_n$ be the sequence of replacements of substrings in S with the substrings in R. Then

Alignment $L_{S:R}$ is a latent sequence like allophone models sequence in HMM-based speech recognition. Taking reasonable upper limit on the length of l_i it is possible to use a modified version of EM-algorithm, similar to *Baum-Welch* [5] as proposed in [1]. Moreover, in place of $\mathbf{P}(l_i)$ a bigram probability $\mathbf{P}(l_i \mid l_{i-1})$ may be taken, thus forming the model of higher order.

We calculate Levenshtein similarity measure as normalized Levenshtein distance LD between candidate and pattern strings:

$$L_1 = \frac{LD}{N}$$

where N is the length of the candidate. To calculate the probability based similarity measure we also should reckon difference of lengths and the fact that $\mathbf{P}(S,S) \neq 0$, which is not true for the valid distance:

$$L_2 = \left| \frac{\log\big(\mathbf{P}(S,S)\big)}{M} - \frac{\log\big(\mathbf{P}(S,R)\big)}{N} \right| + |M - N| \cdot \alpha,$$

where M and N are accordingly lengths of the pattern and the candidate, α is the penalty for difference between string lengths used as a hyperparameter.

The criterion of taking the decision that R is wrong recognition of S was taken if the measures L1 and L2 exceeded corresponding thresholds.

4. Experiments

In our experiments we used the dialog system query log. We chose the queries which system could not respond and extracted those which contained named entities.

The initial sample comtained 15,000 different queries. We took the queries with corpus counts above 20 and excluded TV control queries (poweroff, next channel etc.). Based on the popular queries we automatically obtained regular expressions for the extraction of the candidate strings.

There were 916 pairs named entity-recognized string in the training set and 89 pairs in the test set.

Both algorithms took as an input the list of candidate strings. The output was the list of pairs named entity-recognized string. Based on the lists in the test set and the algorithm outputs we calculated precision, recall and F_1-measure.

The results of the experiments for Levenshein-based algorithm and two versions of the HMM-based algorithm with first and second oreder probabilities are given below:

Method	Precision	Recall	F_1-measure
Levenshtein distance	0.76	0.72	0.74
Unigrams	0.85	0.98	0.91
Bigrams	0.83	0.97	0.89

Poor performance of the bigram model may be the result of overfitting.

5. Discussion

We proposed the algorithm for the gazetteer update based on the users' queries to the dialog engine. The proposed solution is able not only update a gazetteer but also detect different variants of wrong recognitions returned by the external ASR system.

The HMM based similarity measure allows to get more accurate predictions when we try to match some candidate string against patterns in the gazetteer. The key advantage of the method is that due to the use of the probabilistic measures of string transformation the similarities between strings transformed by means of likely replacements are less than for strings transformed by means of rare replacements. The algorithm for HMM-based similarity calculation performs better in terms of F1-measure than widely used Levenshtein distance.

References

[1] *Bisani M., Ney H.,* (2008), Joint-sequence models for grapheme-to-phoneme conversion, Speech Communication, 50 (5), 434–451.

[2] *Lucassen J. M., Mercer R. L.,* (1984) An information theoretic approach to the automatic determination of phonemic baseforms, Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'84, Vol. 9, pp. 304–307

[3] *Luk R. W. P., Damper R. I.,* (1996), Stochastic phonographic transduction for English, Computer Speech & Language, 10(2), 133–153

[4] *Rabiner L., Rosenberg A., Levinson S.,* (1978), Considerations in Dynamic Time Warping Algorithms for Discrete Word Recognition, The Journal of the Acoustical Society of America, 63(S1), S. 79-S79.

[5] *Rabiner L.,* (1989), A tutorial on hidden Markov models and selected applications in speech recognition, Proceedings of the IEEE, 77(2), 257–286.

[6] *Schulz Klaus U.; Mihov Stoyan,* (2002), Fast string correction with Levenshtein automata, International Journal on Document Analysis and Recognition, 5(1), 67–85.