Meta-learning for Fast Dialog System Habituation to New Users

Raimo Bakis[1], Jiří Havelka[2], and Jan Cuřín[2]

[1]IBM T. J. Watson Research Center
1101 Kitchawan Rd, Yorktown Heights, NY 10598, USA
bakis@us.ibm.com

[2]IBM Prague Research Lab
V Parku 2294/4, Prague, 148 00, Czech Republic
{havelka,jan_curin}@cz.ibm.com

Abstract

When a new user comes on board a mobile dialog app, the system needs to quickly learn this user's preferences in order to provide customized service. Learning speed depends on miscellaneous parameters such as probability smoothing parameters. When a system already has a large number of other users, then we show how meta-learning algorithms can leverage that corpus to find parameter values which improve the learning speed on new users.

Motivation

You arrive at the office one day and say to your new assistant:

> *"Get me Joe on the phone, please."*
> The assistant looks at your address book: *"Is that Joe Smith or Joe Brown?"*
> You say, *"Brown."*
> Next morning, again, *"I need to talk to Joe."*
> Assistant: *"Brown, right? Not Smith?"*
> You: *"Yes, Brown."*
> The next time you want Joe, the assistant probably doesn't even ask, maybe just confirms: *"OK, calling Joe Brown."*

An electronic assistant, too, needs to behave in a sufficiently human-like manner to enable the user to develop a sense of mutual understanding (establish *rapport*).

Related Work

In recent years, there has been a great amount of research into statistical dialog systems—Partially Observable Markov Decision Processes (POMDPs), in particular, offer a general framework for implementing a statistical dialog manager (DM) module; see Young et al. [3] for a survey.

We present an approach that allows us to take a step in the direction of incrementally introducing statistical modules into dialog systems, be it existing or newly developed. We concentrate on modeling beliefs over sub-parts of the full dialog state that can be used by hand-crafted or automatically learned policies for sub-dialogs.

Meta-learning through a Pilot Experiment

Task definition: Given a short list of contacts, our goal is to reduce the DM's remaining uncertainty—measured in terms of entropy—about the user's intent to as low a value as possible.

Data Sample

We selected 134 dialog sessions on mobile devices where a user asked to make a phone call but the DM was unable to determine a unique phone number on the basis of the user's initial request; the DM then presented a list of most likely choices to the user, both in a GUI display and by Text-To-Speech. We selected only those dialogs in which the user subsequently made a unique selection from that list and where the DM then dialed that number—on the assumption that it was the user's true intent for the dialog. (The total sample we considered consisted of 2,790 dialog sessions.)

Formulation as Optimization Problem

We formulate the task as an optimization problem, which allows us to find optimal values of parameters of a statistical algorithm of our interest using standard numerical optimization methods.

Let K be the number of dialogs; let I_k be the length of the contact short list in the k-th dialog, where $k = 1, 2, \ldots, K$.

Now consider the contact who appears in position i in the short list in the k-th dialog. Let $n_{k,i}$ be the total number of times this contact has already been called by the same person who is the user in the k-th dialog. Furthermore, let i_k be the index of that user's true intent in the short list in this, the k-th dialog.

Consider a statistical algorithm that calculates a belief ("subjective" probability) $b_k(i)$ for the i-th contact in the short list in the k-th dialog; these values are normalized so that

$$\sum_{i=1}^{I_k} b_k(i) = 1 \quad \forall\, k \in \{1, 2, \ldots, K\}.$$

Now let $p_k(i)$ designate the true probability of the i-th candidate in the k-th dialog. Because we assume we know the user's actual intent with certainty, we have $p_k(i_k) = 1$ and $p_k(j) = 0$ for all $j \in \{1, 2, \ldots, I_k\}$ such that $j \neq i_k$. In that case, we obtain the following for the cross-entropy between p_k and b_k

$$\begin{aligned} H(p_k, b_k) &= -\sum_{j=1}^{I_k} p_k(j) \log b_k(j) \\ &= -\log b_k(i_k). \end{aligned}$$

We now define our *objective function* as the average value of that cross-entropy:

$$F = -\frac{1}{K} \sum_{k=1}^{K} \log b_k(i_k). \tag{1}$$

Clearly, if the beliefs were always correct, so that

$$b_k(i_k) = 1 \quad \forall\, k \in \{1, 2, \ldots, K\},$$

then F would achieve its minimum value, $F = 0$. In practice, we will generally not reach that value, but our aim is to minimize F.

Whenever we can evaluate the gradient of an objective function with respect to its parameters, we can try to find optimal values of the parameters using standard numerical optimization methods, such as gradient descent or L-BFGS (Liu and Nocedal [2]). In our case, we would tune any relevant parameters of a statistical algorithm implementing belief estimation of our choice.

A Concrete Algorithm — Laplace Smoothing

When calculating $b_k(i)$, a general statistical algorithm can, in principle, use all information in existence at the time of that computation. In this pilot study, we restrict the computation to use only the counts $n_{k,i}$ and one additional parameter α. We denote this by writing

$$b_k(i \mid n_{k,1}, \ldots, n_{k,I_k}, \alpha).$$

When, for some k, all counts are large, that is $n_{k,i} \gg 1$ for all $i \in \{1, 2, \ldots, I_k\}$, then it might be reasonable to use the maximum-likelihood estimator for all $i = 1, 2, \ldots, I_k$

$$b_k(i \mid n_{k,1}, \ldots, n_{k,I_k}, \alpha) = \frac{n_{k,i}}{\sum_{j=1}^{I_k} n_{k,j}}. \tag{2}$$

But we are interested in hitting the ground running, so to speak. Our probability estimates must be as accurate as possible even when $n_{k,i} = 0$ for some k and i. In those cases equation (2) would clearly be wrong, because it would set $b_k(i) = 0$ whenever $n_{k,i} = 0$. This would imply that the DM knew with certainty that the user was not calling candidate i in dialog k, that is, it would have to assume that it mis-heard, or that the user mis-spoke.

But if equation (2) is wrong, then what is the true probability of calling somebody new, somebody this user has never called before with this app? That question—how to calculate the probability of a hitherto unseen event—has vexed philosophers for centuries. What is the probability that the sun will not rise tomorrow? Or the probability of a black swan? Fortunately, in our application, we don't have to philosophize or speculate, we can answer the question empirically through meta-learning.

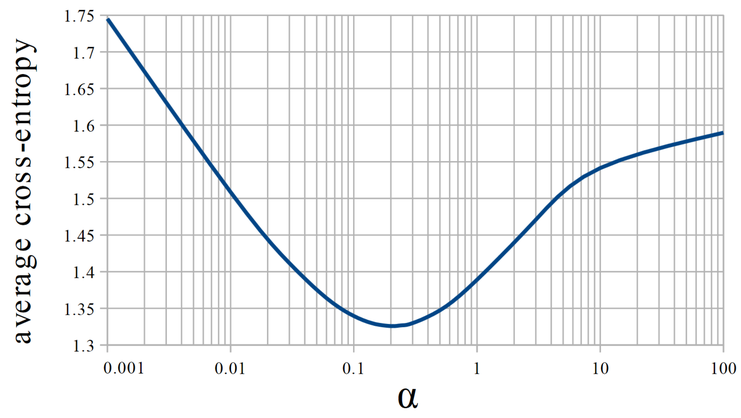
A frequently used method for getting rid of the zero probabilities for unseen events is probability smoothing. Perhaps the simplest and oldest way of doing this is *additive smoothing*, also known as *Laplace smoothing*. This method simply adds a "smoothing constant" α to each observed count $n_{k,i}$. The probability estimate for candidate i in the k-th dialog becomes then for all $i = 1, 2, \ldots, I_k$

$$b_k(i \mid n_{k,1}, \ldots, n_{k,I_k}, \alpha) = \frac{n_{k,i} + \alpha}{\sum_{j=1}^{I_k} (n_{k,j} + \alpha)}. \tag{3}$$

Under certain theoretical assumptions, the correct value for the smoothing constant would be $\alpha = 1$. In realistic situations, however, this is generally not the optimal value; that is why it is necessary to determine α empirically. Within any given population of users there is, of course, some variation of behaviors, but we can at least determine the best value for the population average.

Results

The figure shows how the average cross-entropy defined by equation (1) depends on α in our sample. For this population of users we found the optimal value to be $\alpha = 0.205$. This is significantly lower than the sometimes-quoted theoretical value of 1, but is definitely larger than zero, which vindicates that meta-learning can indeed improve belief estimation for new users using data from a population of other users.



Future Work

The beliefs calculated from equation (3) are the same in the morning as in the evening. To accommodate the more general case where the frequency of you calling your contacts might depend on the time of the day, we could let the values $n_{k,i}$ in equation (3) be functions of the time of day. Let φ be the time-of-day phase of the proposed new call: 0 at midnight, 2π next midnight; then we might write

$$n_{k,i}(\varphi) = \sum_{m=0}^{M} \left(\alpha_{m,k,i} \cos m\varphi + \beta_{m,k,i} \sin m\varphi \right). \tag{4}$$

The α and β coefficients would be computed by some algorithm from not only the total number of previous calls to the same contact by the same caller, but also from the times of these calls; algorithms related to Fourier analysis are obvious candidates. These algorithms would involve parameters affecting smoothing across candidates as well as smoothing across time, which could be optimized analogously as above.

The predicted beliefs could be allowed to depend on many other variables as well, in addition to the time of day. Obviously, a user's preferences change as time passes, which could be modeled using a decay parameter (a function of elapsed time) applied to individual past observations. The user's geographic location at the time of the call may predict which pizza shop gets the order. The probability of calling a taxi service may depend on the weather, etc.

When the beliefs are calculated as functions of quantities for which we have good intuitive models, then we can design special-purpose functions such as those in equations (3) and (4). When this is not possible, then more general families of parametric functions may be appropriate, such as artificial neural nets (see for example Bishop [1]).

Summary

Cognitive behavior on the part of a mobile device—behavior which appears thoughtful and intelligent to the user—requires, among other things, that the system anticipate users' wishes and intents as quickly as possible. How fast and how accurately it learns a new user's preferences depends on parameter settings which, in turn, must be learned from a sample of other users from the same population. We illustrate this principle by means of a simple probability-smoothing example, and sketch extensions of the algorithm for handling more complex patterns.

References

[1] Bishop, C.M. (2006) *Pattern Recognition and Machine Learning*. Springer

[2] Liu, D.C., Nocedal, J. (1989) On the limited memory BFGS method for large scale optimization. *Mathematical Programming* 45(1-3):503–528

[3] Young, S., Gašić, M., Thomson, B., Williams, J. (2013) POMDP-Based Statistical Spoken Dialog Systems: A Review. *Proceedings of the IEEE* 101(5):1160–1179