Aligning Real-Time Opinion Poll Responses with an Expectation-Maximization Algorithm

Isaac Julien
Department of Computer Science
University of Maryland
ijulien6@gmail.com

Philip Resnik
Department of Computer Science
University of Maryland
resnik@umd.edu

Abstract

React Labs [8], a mobile polling application, makes it possible to analyze individuals' real-time reactions and opinions on a large scale. While users tend to respond nearly instantaneously to events as they see them [4], in the setting of a televised debate users may actually be viewing the same event at different times. As a result, the timestamps attached to responses collected using React Labs may not align correctly with a reference such as a transcript of the debate.

We propose a generative model for this problem in which each user responds to latent events at a fixed scalar offset, also latent, and we derive an Expectation Maximization (EM) algorithm to estimate both the times of events and the offsets of users.

We evaluate the method using a corpus of reactions obtained during the first 2012 presidential debate [3]. Since the true offsets of the users in the real data are unknown, we create artificial data with known offsets, add error and omissions, and place these alongside real data. The EM algorithm recovers the offsets accurately, suggesting that this method may be used to estimate offsets, which can then be used to adjust response timestamps and create a modified, more accurately aligned dataset.

1 Introduction

1.1 React Labs

React Labs [3] is a mobile application platform that allows users to respond in real time to live events. Collecting these responses provides an opportunity for large-scale political and social analysis. Its applications include political polling, in which participants view-

ing televised and live-streamed debates can respond to a speaker's statements by selecting one of a number of preset reactions.

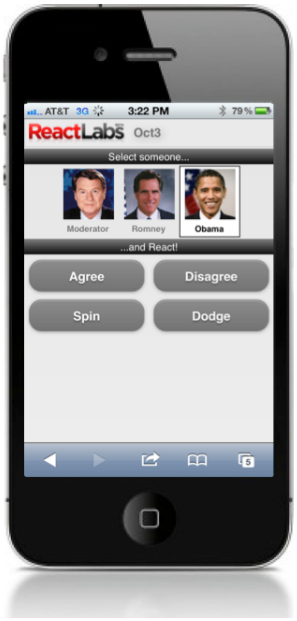

Figure 1: React Labs: Educate screenshot

During the second 2012 presidential debate between Barack Obama and Mitt Romney, React Labs: Educate [8] collected more than 200,000 responses from more than 3,000 users watching on television or streaming online. The application's interface shows the users each person involved in the debate (including the moderator), and four reactions: "Agree," "Disagree," "Spin," or "Dodge." The user selects a person and a reaction to respond as shown in the screenshot in Figure 1. For example, a user may tap "Obama" and then "Agree" to indicate agreement with a statement Obama has just made. Additionally, the user can respond to survey questions to provide information such as age, gender, and political views.

The collected responses can then be used to study what prompts certain types of reactions. For example, (Boydstun, Glazier, Pietryka, Resnik) finds that users who placed a higher priority on economic issues tended to disagree with Romney's statements regarding the economy, and that Republicans tended to disagree most with Obama when he discussed foreign affairs [3].

1.2 Broadcast Delays

Much of the analysis that having this real-time data makes possible is dependent on the accuracy of the timestamps attached to the reactions, since these link the reactions to their causes. One possible source of error is the delay between when a user experiences a reaction to some event and when the user can physically register this response. However, studies suggest that this delay is minimal [4].

A more significant source of error is the existence of different delays for local television broadcasts, even for a nationally broadcast event on the same television network. Although the application will record the time of a response accurately, the responder might be watching at a non-trivial offset from other responders, sometimes as much as 5 or more seconds. High-definition cable broadcasts also tend to lag slightly behind standard-definition broadcasts.

Although this almost certainly causes inaccuracy in the collected responses, measuring how much is difficult. Trying to correct the

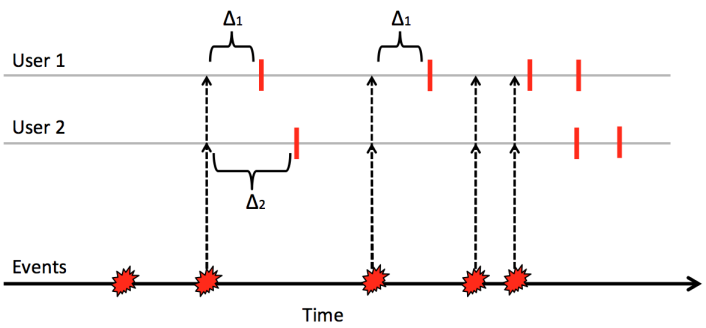

Figure 2: Modeling events, delays, and reactions

timestamps is then a very challenging task, because both the offsets and the causes of the reactions are unknown.

1.3 Related Work

This project is motivated by the application of React Labs to political analysis described in (Boydstun, Glazier, Pietryka, Resnik) [3].

(Listgarten, Neal, Roweis, Emili) [5] introduces a Continuous Profile Model for aligning and relating multiple continuous time series, trained with an EM algorithm. This motivated in part the EM approach for aligning React Lab responses.

Some other work has addressed similar problems involving learning hidden events and causation. (Wingate, Goodman, Roy, Tenenbaum) [6] introduces the Infinite Latent Events Model, which infers latent events and causations between events from discrete time-series data. (Simma, Jordan) [7] models cascades of events with Poisson processes.

Much work has also been done to understand delays in data networks. This paper treats the television network as a black box that causes delays independent of any user-specific information, and incorporating any knowledge of network delays is left for future work.

1.4 Data

We use a set of 172,045 responses collected from 3308 users during the second 2012 presidential debate. Responses from participants who watched streaming online were first discarded to focus on television network-induced delays, since streaming delays may be much longer.

2 Motivation

2.1 The Alignment Problem

Let us use the term "reaction-worthy event," or simply "event," to refer to statements made in the debate that a user may respond to. We make three simplifying assumptions to approach the problem of aligning responses to unknown events. First, we assume that each of a given user's responses is delayed from the true time of the event by a constant amount. This means that each user can be assigned an offset, and if that offset is known then the true times of the responses are also known. Figure 2 portrays this model.

Second, we assume that the range of offsets is on the order of several seconds. In our experiments we use a window of possible offsets between -5 and 5 seconds. This window was chosen by analyzing a set of 20 local broadcasts for one of the 2012 presidential debates in order to assess the variability of local broadcast delays.

Finally, we assume that many of the unseen events cause multiple users to react. In the context of a political debate, it seems reasonable that certain events, for instance, a candidate's statement of a position on a certain issue, might cause reactions from many of the responders.

We hypothesize that several users may all respond to a subset of the more reaction-worthy events, and that this may be used to infer the true time of the event and align their responses. For example, two users U1 and U2 may record reactions at times [10, 20, 30, 40] and [11, 21, 31], respectively. If we shift the responses of U2 by an offset of -1, the data looks much more likely under our assumption. Our approach is a formalization of this idea.

2.2 Simple MLE Alignment

Our first approach to alignment estimates only the offsets and ignores the estimation of where the hidden events occur.

We treat the debate as a series of discrete time steps $[1, ..., T]$. At each point t, some total number of responses from all users is observed. One can interpret the density of responses at t as the probability that any user responds to an event at t. Call the size-T array of these probabilities P. The algorithm is shown in Figure 3.

Calculate P and smooth using Gaussian window
Repeat until no new offsets are assigned:
  For each user u:
    Let Δ_u be a possible offset for u
    Let R_u be set of observed responses for u
    $\Delta_{NEW} = \arg\max_{\Delta_u} \prod_{t \in R_u} P[t + \Delta_u]$
    Assign offset Δ_{NEW} to u
    For each $t \in R_u$:
      $t < -t + \Delta_{NEW}$
    Recalculate and resmooth P

Figure 3: Simple alignment algorithm

To calculate the density of responses, we need count the number of reactions at each t. In reality, each reaction can fall into one of several categories (Obama: Agree, Romney: Agree, etc.) as in Figure 1. For now, this is ignored, and all reaction types are treated as equivalent.

2.3 Evaluating Alignment

Evaluation on real data is challenging because the true offsets of users are not known, nor are the times of the events that caused the reactions. However, it is reasonable to expect that within the dataset, there are subsets of users who respond similarly. To evaluate how well similar sets of responses might be aligned, we can duplicate certain sets of responses, assign each duplicate an artificially generated (and therefore known) offset, and then measure how well the algorithm recovers these artificial offsets.

D users are chosen at random from the set of all users. Each is "cloned" K-1 times each, creating K sets of reactions for each of the D users, a total of D*K known offsets. For the cloned data to be realistic, it is important that they not be perfect copies of the original D users. Therefore, when a set of responses is copied, each observation is omitted with some probability p_{omit}, and $[0, err]$ seconds of error is added uniformly at random to each included observation. The offsets for the cloned users are generated uniformly at random in the range [-5, 5], and the offsets for each of the D users from which clones are generated are assumed to be 0. The cloned data are then included with data from N other distinct users, also selected at random from the dataset.

The offsets Δ_u for every user u are then estimated with the alignment algorithm. Using the data "as-is," without calculating offsets, is equivalent to estimating $\Delta_u = 0$ for all u. We therefore measure our estimate of Δ_u against a baseline of $\Delta_u = 0$.

We use two methods for evaluating the quality of the estimated offsets. The first is the MSE (Mean Squared Error) of the estimated offsets compared to the true offsets. The second is the accuracy with which offsets are predicted exactly.

2.4 Results

Table 1 shows the average MSE and Accuracy of the estimated offsets over 10 runs, for different values of D, K, and p_{omit}.

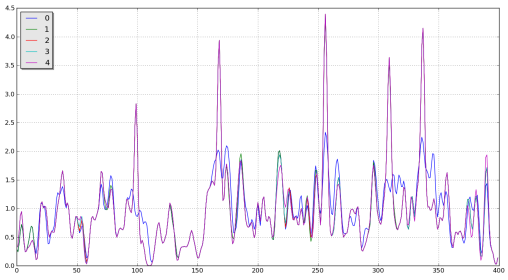

Figure 4: Distribution of event probabilities iterations 0-4

Figure 4 displays the probabilities at each E[t] over 5 iterations using the estimated offsets at each iteration. The sharp spikes after the final iteration show where the cloned responses have been aligned.

While simple, this approach performs well enough to motivate a more thorough model and analysis in the following sections.

D	K	p_{err}	p_{omit}	Avg. MSE Baseline	Avg. MSE	Avg. Acc Baseline	Avg. Acc	**Reduction in MSE**	**Reduction in Error**
20	2	1	0.4	4.965	0.545	0.993	0.780	**79.1%**	**51.8%**
100	2	1	0.4	5.128	0.540	0.111	0.996	**97.8%**	**99.1%**
100	5	1	0.4	7.958	0.273	0.1000	0.997	**98.7%**	**99.6%**
100	2	1	0.6	4.992	0.546	1.369o	0.955	**72.3%**	**89.9%**

Table 1: Recovering offsets for duplicated data: N=1000, 10 iterations

U	Set of all users
T	Number of discrete time points in the debate
u	User u
p	Probability of an event at any t
$E[t] \in 0, 1$	1 if an event occurs at time t
p_u	Probability that user u reacts to an event
Δ_u	Offset for user u
$R_u = \{t_1, t_2, ..., t_k\}$	Observed reactions for user u
σ^2	Parameter of Normal Distribution

Table 2: Definitions

3 An EM Algorithm for Alignment

For $u \in U$:
  Choose $\Delta_u \sim Normal(0, \sigma^2)$
For $t = 0, 1, 2, ..., T$:
  Choose $E[t] \in \{0, 1\} \sim Bern(p)$
    For $u \in U$:
      Choose $r \in \{0, 1\} \sim Bern(E[t] * p_u)$
      If $r == 1$:
        Add $r + \Delta_u$ to R_u

Figure 5: Generative Process

3.1 Model

We develop the following generative model for the response data using the definitions in Table 2:

As before, we interpret the debate as a series of discrete time points. At each time t, a reaction-worthy event occurs with probability p. Each user u responds to this event with probability p_u. If the user responds, then a reaction is recorded at time $t+\Delta_u$ in order to account for the offset Δ_u. The offsets are gen-

erated from a Normal distribution with zero mean.

The sets of reactions R_u are the only observable variables. Each Δ_u and $E[t]$ is unobservable, and p, σ^2, and each p_u are parameters of the model.

3.2 Expectation Maximization

Given a probabilistic model involving both hidden variables and unknown parameters, an Expectation-Maximization algorithm maximizes the likelihood the model by iteratively estimating the values of the hidden variables and the parameters [1].

In the E-step, the algorithm computes a distribution Q over the values of each hidden variable Z, given the values of the observed variables, X [2]:

$$Q(Z = z) = p(Z = z|X; \Theta) \tag{1}$$

Next, new values for the parameters are estimated in the M-Step:

$$\Theta = \arg\max_{\Theta} \sum_{Z} \sum_{Z=z} Q(Z = z) log(\frac{p(x, z; \Theta)}{Q(Z = z)}) \tag{2}$$

That is, we choose the set of parameters Θ that maximize the log likelihood of the observed data, accounting for each possible setting z of random variable Z proportionally to $Q(Z = z)$.

3.3 E-Step and M-Step Updates

For the model described in Figure 5, the E step must calculate the distribution over the possible values of each Δ_u and of each $E[t]$. For each Δ_u :

$$Q(\Delta_u = \delta) \propto P(R_u) * P(\overline{R_u}) * P(\delta) \tag{3}$$

$$P(R_u) = \prod_{t \in R_u} p_u E[t - \delta] \tag{4}$$

$$P(\overline{R_u}) = \prod_{t \notin R_u} (1 - p_u)E[t - \delta] + (1 - E[t - \delta]) \tag{5}$$

$$P(\delta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{\delta^2}{2\sigma^2}} \tag{6}$$

Eq. 3 states that the (posterior) probability of setting δ for offset Δ_u is proportional to the product of three probabilities: The probability of observing each reaction in R_u (Eq. 4), the probability of *not* observing reactions at other times (Eq. 5), and the probability of offset δ under the Normal distribution (Eq. 6). The current estimates of the $E[t]$s are used for calculating $E[t - \delta]$ in Eq. 3-4.

Each $E[t]$ must then be calculated using the distributions over the Δ_u already computed. If we could treat each E[t] as independent, then this calculation would be simple:

$$\begin{aligned} Q(E[t] = 1) &\propto \prod_{u \in U} p(R_u|E[t]) * p \\ Q(E[t] = 0) &\propto \prod_{u \in U} p(R_u|E[t]) * (1 - p) \end{aligned} \tag{7}$$

However, this is not the case. Consider an example in which two reactions from users u_1

and u_2 are observed at t=100 and 101. This can be explained by the model in one of two ways: (A) Both reactions are caused by separate events and u_1 and u_2 have the same offsets, or (B) A single event caused both reactions, and the offsets of u_1 and u_2 differ by 1. If p is relatively low, then (B) is far more likely. So when calculating the distribution over settings of E[100] we must also look at the possible settings of E[101].

This applies to all E[t], so without the assumption of independence, we must calculate the distribution over settings of a single variable E with an exponential number of settings.

Since this is intractable, we instead approximate each E[t] individually based on the observed reactions. We allow each reaction from every user to contribute a "vote" for an event at time t proportional to the probability of the offset that would have been necessary to observe the reaction based on an event at t and the previous estimate of E[t].

$$Q(E[t]) \propto \sum_{u \in U} \sum_{\Delta_u = \delta} \sum_{\substack{t_r \in R_u \\ t_r + \delta = t}} Q(\Delta_u = \delta) + E[t] \quad (8)$$

The $Q(E[t])$ are then scaled to match the current estimate of p. Using this approximation, re-estimating p and each p_u does not make sense, so the algorithm is in effect stuck with the initial estimate of both.

We choose $\sigma^2 = 25$ so that all possible offsets within the window of [-5, 5] that the algorithm considers are within one standard deviation of the mean (0).

4 EM Experiment

4.1 Recovering Offsets of Synthetic Data

To verify that the algorithm is working, we generate several sets of synthetic data according to the model defined in Figure 5. The estimate of p is the average number of responses per user over T, and the estimate of each p_u is the number of responses from user u over T. σ^2 is fixed at 25.

In all test runs the offsets and events were successfully recovered, and the per-user response probabilities p_u were estimated accurately. This provides assurance that the algorithm works as desired and that the approximation of $E[t]$ is reasonable.

4.2 Evaluation on Cloned Data

We then evaluate the algorithm using cloned data as in section 2.3. After running EM, we select the maximum likelihood estimate of Δ_u for each user as the estimated offset.

Table 3 displays the results for several experiments with different parameters. Figures 6 and 7 show the change in MSE and accuracy over iterations for several representative runs of EM. Figure 8 shows several sets of reactions as they are aligned in EM.

D	K	p_{err}	p_{omit}	Avg. MSE Baseline	Avg. MSE	Avg. Acc Baseline	Avg. Acc	**Reduction in MSE**	**Reduction in Error**
20	2	1	0.4	4.695	0.546	0.693	0.958	**85.6%**	**91.1%**
100	2	1	0.4	5.181	0.539	0.024	0.997	**99.6%**	**99.3%**
100	5	1	0.4	8.114	0.273	0.0298	0.997	**99.6%**	**99.6%**
100	2	1	0.6	4.862	0.548	0.292	0.976	**94.0%**	**94.7%**

Table 3: Recovering offsets for duplicated data with EM: N=1000, 10 iterations

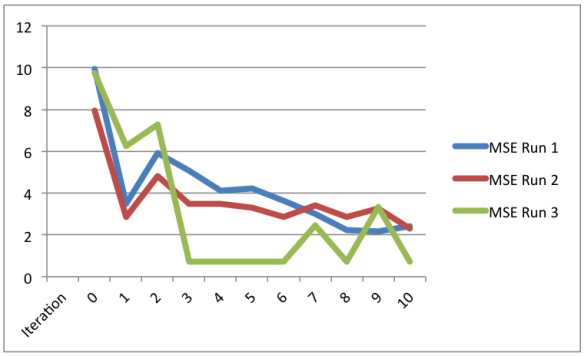

Figure 6: MSE over iterations 1-10

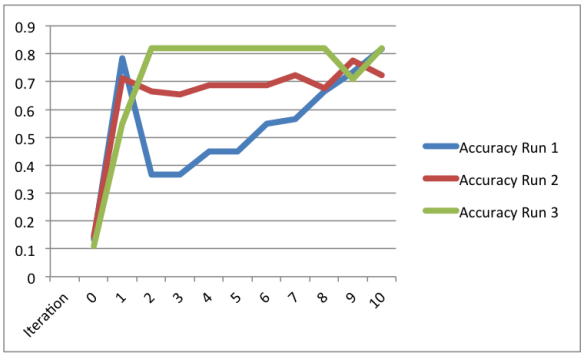

Figure 7: Accuracy over iterations 1-10

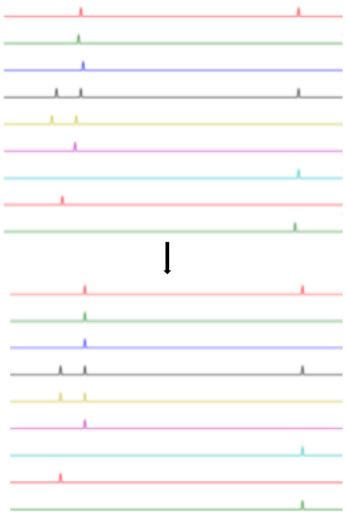

Figure 8: Sample of reactions before and after alignment

5 Discussion

5.1 Results

The EM algorithm is consistently able to recover offsets for artificial but realistic users

with high accuracy. As opposed to using the unmodified original data, accounting for these offsets almost entirely eliminates errors in reaction placement as determined by MSE and accuracy (Figure 3).

The simpler model discussed in Section 2 works surprisingly well, but EM performs better in general and is more resilient to omission. With the omission probability set to 60%, EM achieved a 94% reduction in MSE, while the simpler model recorded a 72.3% reduction.

We have applied the algorithm to the full dataset to obtain an offset for each user included in the 2012 presidential debate corpus. These offsets will be used to adjust the timestamps of each reaction and to improve the accuracy of analysis done with the dataset [3].

5.2 Future Work

The generative process described in Figure 5 is simple and presents several possibilities for expansion. One possibility would be to incorporate information from the debate transcript, such as which candidate is speaking at time t, and relate this to user features such as political affiliation and reaction types. For instance, it is probably more likely that an Obama supporter responds "Obama:Agree" to a statement made by Obama than to a statement made by Romney a few seconds later. One could also include a more comprehensive model of the processes by which delays are generated, and attempt to align responses from users watching online streams. For a more complex model, a Gibbs sampling approach may be more appropriate.

Finally, extensions of this model could be used to investigate interactions between users' issue priorities, the candidate speaking, and what the candidate is saying, using manual encoding of topics and framing [3], textual context, or a combination of the two.

References

[1] A. P. Dempster, N. M. Laird, D. B. Rubin *Maximum Likelihood from Incomplete Data via the EM Algorithm.* Journal of the Royal Statistical Society. Series B (Methodological), Vol. 39, No. 1. (1977), pp. 1-38.

[2] Andrew Ng *The EM algorithm.* Stanford University http://cs229.stanford.edu/notes/cs229-notes8.pdf

[3] Amber E. Boydstun, Rebecca A. Glazier, Matthew T. Pietryka, Philip Resnik *Real-Time Reactions to a 2012 Presidential Debate: A Method for Understanding Which Messages Matter.* Public Opion Quarterly, in press.

[4] Amber E. Boydstun, Rebecca A. Glazier, Matthew T. Pietryka, Philip Resnik *Real-Time Reactions to a 2012 Presidential Debate: A Method for Understanding Which Messages Matter.* Public Opion Quarterly, in press. Supplemental Materials

[5] Jennifer Listgarten, Radford M. Neal, Sam T. Roweis, Andrew Emili *Multiple alignment of continuous time series.* Advances in Neural Information Processing Systems. MIT Press. 2005, pp. 817-824

[6] David Wingate, Noah D. Goodman, Daniel M. Roy, Joshua B. Tenenbaum. *The Infinite Latent Events Model.* Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI '09). AUAI Press, Arlington, Virginia, United States. 2009, pp. 607-614.

[7] Aleksandr Simma, Michael I. Jordan. *Modeling Events with Cascades of Poisson Processes.* Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence (UAI2010). 2012.

[8] *React Labs.* http://reactlabs.wordpress.com/