Modeling 802.11 AP Usage through Daily Keep-alive Event Counts

Dossa Massa
Faculty of Engineering and INESC Porto
University of Porto
Porto, Portugal
mmohamed@inescporto.pt

Ricardo Morla
Faculty of Engineering and INESC Porto
University of Porto
Porto, Portugal
ricardo.morla@fe.up.pt

***Abstract*—Wireless and in particular 802.11 is one of the major technologies for accessing the Internet at home, in coffee shops or other public places, and in enterprises and university campuses. While most recent work on modeling wireless sites focuses on user mobility, this paper presents and compares a number of models for characterizing access point (AP) usage; moreover, rather than looking at throughput we focus on daily counts of keep-alive events that mobile devices generate every 15 minutes they are connected to the wireless network. Our models are trained and evaluated on data collected from a Porto hotspot of Eduroam, the European academic wireless network. The models we present are generative, in the sense they can be used to generate synthetic daily event counts for a single AP or a collection of APs. We provide standard cross-validation comparison of models using the log-likelihood of the models on training and test data.**

802.11; AP usage modeling; Log-likelihood; Gamma mixture of exponentials; Table CPD.

I. INTRODUCTION

Wireless LANs with 802.11 technologies are increasingly common in most universities and organizations, with entire cities already having such infrastructures [1]. Efforts in trying to understand the usage of these wireless networks started some years ago, namely at Dartmouth [2], looking first at the monitoring infrastructure and at characterizing the global properties of the network usage and then focusing on user mobility. Understanding the usage of these networks is critical not only for better understanding the evolution of the adoption of this technology, but also for network design, planning, and maintenance, and for evaluating new protocols and applications that run on these networks.

The emphasis of this paper is on devising and comparing generative models of Access Point (AP) usage that can be used to drive the evaluation of new protocols and applications, for example in network simulators. The variables we want to model in this paper are AP daily keep-alive event counts. A keep-alive event is a message sent by a mobile client every 15 minutes for refreshing the client's association with an AP. Because of their periodic nature, keep-alive statistics tell us about the time users are associated with an AP, during which they may generate traffic with different profiles. Rather than modeling traffic directly, we chose to model event counts given: 1) the evidence that user-AP association and thus event counts are correlated with traffic [7]; 2) the possibility of plugging-in different traffic models onto our event count models to generate different network traffic patterns according to the needs of simulation experiment and application or protocol to evaluate. In this paper we focus on the event count model and use the total number of events per day per AP as both training and test data for our models.

We proceed as follows. In section II we review related work on 802.11 usage modeling. In section III we describe the data set that we use for training and test as well as the log-likelihood function that we use as metric for comparing models on the same data set. The description of our modeling activities starts in section IV by considering the case of APs which have all-zero daily event counts. We assume these are mis-configured APs and we use a Bernoulli distribution as a pre-model for removing these APs from the other models. Our models that do not consider dependencies in time are described in section V. These include a simple exponential model, a discrete mixture of exponentials, a continuous Gamma mixture of exponentials, and an above-below model that allows thinking about the modeling problem in binary terms. In section VI we present a table conditional probability distribution model that uses binary variables for predicting the next value for an AP given its T-1 previous samples. Because this model uses binary variables, we cannot compare its log-likelihood directly with previous models. To do this comparison, we integrate it with the above-below model of the previous section. For each of the previous models we provide their log-likelihood on a specific training and test set. In section VII we employ cross-validation in order to increase the accuracy of our models and of their evaluation. Finally in section VIII we present concluding remarks.

II. RELATED WORK

There are many studies on wireless 802.11 usage. Most, however, focus on modeling user behavior. For example, W. Husu et al [3] analyze 94 days of Dartmouth 802.11 traces. They aggregate APs within a building and consider the number of users in the building during 24 hour slots. They observed hundreds of distinct user groups with commonalities in their usage patterns which follow power law distributions. Another work by M. Boc at el. [4] studied the individuality of users' mobility patterns from Dartmouth traces for two months. Without using geographical information they proposed a mobility aware clustering algorithm that uses the number of roaming events as a metric to approximate proximity of APs. Through their algorithm they were able to depict individuals' mobility as well as to

map clusters to places with social meaning and paths. D. Kotz et al. [5] analyzed Dartmouth traces over a two year period. They presented results of the empirical evaluation of user location predictors using four major families of domain-independent predictors, namely Markov-based, compression based, Prediction by Partial Matching (PPM) and Statistical Parametric Matching (SPM) predictors. They found that low-order Markov predictors performed better than more complex and more space-consuming compression-based predictors.

The focus of our paper is not on modeling user mobility and identifying user mobility patterns, but on probabilistic models for AP usage. Although the two problems are related, the emphasis on AP usage and the use of probabilistic generative models is better suited for application to network simulation.

Focusing on the infrastructure rather than on the users, [6] tried to understand the usage patterns of an 802.11 campus wireless LAN when it was first deployed and later when it matured, with an emphasis on characterizing active APs and global and application-specific traffic. F. Hernandez et al. [7] tried to characterize the workload of wireless 802.11 APs of two campus-wide infrastructures. They observed similarity in characteristics between the two wireless networks and concluded that log normality is prevalent in both campuses. They also observed correlations between the number of associations and the total traffic per AP.

Despite these modeling efforts, to the best of our knowledge ours is the first attempt to derive generative probabilistic models of AP usage based on daily keep-alive counts and to compare them using the log-likelihood figure of merit on data from the Eduroam network.

III. Experimental Setup

A. Data Set

Our data set is a 183x53 matrix consisting of daily keep-alive event counts for 183 access points in one of the busiest Eduroam hotspots in the University of Porto, on 53 consecutive days. Daily event counts are integers that range from 0 to 2229 (in this data set). We split the data set and used 2 subsets for training/testing: the training set with data from the 53 days of 143 randomly selected access points; and the test set with data from the 53 days of the remaining 40 access points that were not used in the training set.

B. Figure of Merit

For each probabilistic model that we propose in the next sections we compute the log-likelihood of that model on training and test sets. The log-likelihood of a model's probabilistic density function M with parameter Θ on a data set $X = (x_1, x_2, \dots, x_N)$ is defined as:

$$LL(M;\Theta;X) = \sum_{i=1}^{N} \ln M(x_i;\Theta)$$

This is a standard figure of merit in probabilistic learning. For the same data set, better fitting models have higher log-likelihoods. Computing the log-likelihood of different models on the same training data provides an indicator of which model better fits the data that was used to select and train the models. Computing the log-likelihood on the test data provides an indicator of which model performs better on the same, yet unseen data.

IV. All-Zero AP Pre-Model

As with all modeling efforts, considering all-zero entries is a challenge. In our case, this means considering APs for which the training data consists of zero event counts only. We address this challenge by considering a pre-model that we apply beforehand to all our subsequent models when generating synthetic data. We also adjust the log-likelihood of the model on the training and test data to consider this pre-model.

To generate synthetic data using this pre-model, we first draw from a Bernoulli distribution with parameter θ_{zeros} whether an AP is all-zeros or not. If it is, then we generate the number of zero samples we need for that AP. Otherwise we use one of the subsequent models to generate the samples we need for that AP.

We estimate θ_{zeros} by dividing the number of all-zero APs in the training data by the total number of APs in the training data. Of course, this fitting is biased by the duration of the training data, but we believe this issue of all zero APs is not a matter of usage but of AP mis-configuration. Also, we add a "fictional" 1 to the number of all-zero APs in this estimation; this is standard practice in statistical learning and accounts for prior information on the possibility of the existence of all-zero APs even if there are none observed in the training data.

We will not include this pre-model in the definitions of the rest of the models in order to simplify their notation. Nonetheless, we do apply the pre-model to all other models and include its effect on their log-likelihoods. Generically, the PDF and log-likelihood of a data set X in an AP for any of the other models is:

$$p_{\text{model with zeros}}(X;\theta_{zeros},\Theta) = \begin{cases} \theta_{zeros} & : X = 0 \\ (1-\theta_{zeros})p_{model}.(X;\Theta) & : X \neq 0 \end{cases}$$

$$LL_{\text{model with zeros}} = N_{zeroAPs}\ln(\theta_{zeros}) + (A - N_{zeroAPs})\ln(1-\theta_{zeros}) + LL_{model}$$

Where A is the total number of APs, $N_{zeroAPs}$ the number of APs with all-zero event counts in the data set we're evaluating the log-likelihood for, and θ_{zeros} is obtained from the training data. In our training set there are 3 all-zero APs out of 143, which puts our estimate for θ_{zeros} at 2.7778E-2. In the test set we have no all-zero APs. The increase in log-likelihood on this test set that we incur by considering this pre-model with the others is -1.1268.

V. Time-Independent Models

We first consider models that assume the independence between consecutive daily event counts. Although this may not be the case, it caters to simpler models. Later in the paper

we compare these models with others that do consider dependency between consecutive daily event counts.

A. *Exponential*

We start our modeling endeavor by assuming that all daily event counts come from the same distribution, regardless of day or access point. After visualizing the histogram of the training data we picked an exponential distribution and fit it to the training data using a maximum likelihood estimation function in Matlab. The following are the PDF and log-likelihood functions for this distribution:

$$p_{exp}(x;\ \lambda) = \lambda \exp(-\lambda x)$$

$$LL = N \ln \lambda - \lambda \sum_{i=1}^{N} x_i$$

The average daily event count that results from fitting to the training data is 132.53 keep-alive events. The log-likelihoods for this model on the training and test sets are $LL_{train} = -4.3681E4$ and $LL_{test} = -1.2161E4$. These values are only useful when comparing with other models on the same data sets.

B. *Discrete Mixture of Exponentials using K-Means*

A quick analysis of a per-AP daily event count average histogram shows what appears to be a highly skewed, non-normal distribution. As expected from the central limit theorem, this points to the training data for different APs not originating from the same distribution. To consider this observation, we use the training set of AP averages throughout the paper as input for parameter fitting; actually, we use the AP rate (one over the average). This can be justified by thinking that samples from the same AP are to some extent related. Using this information to improve our models without considering time is left for future work.

Here we consider a mixture of distributions and, in particular, a discrete mixture of two or more exponential distributions, which has the following standard mixture PDF and log-likelihood functions:

$$p_{dis\_mix\_exp}(x; W, \Lambda) = \sum_{k=1}^{K} w_k \lambda_k \exp(-\lambda_k x)$$

$$LL = \sum_{i=1}^{N} \ln \sum_{k=1}^{K} w_k \lambda_k \exp(-\lambda_k x_i)$$

Each component of the mixture has a weight w_k, where $(w_1, w_2, \dots, w_K) = W$ and $\sum_{k=1}^{K} w_k = 1$, and a parameter λ_k of the exponential, where $(\lambda_1, \lambda_2, \dots, \lambda_K) = \Lambda$. The standard approach to generate a daily event count sample x using this model is straightforward: first choose a component k by generating a sample from a multinomial distribution with probabilities W. Then generate a value x from an exponential distribution with parameter λ_k.

We used Matlab's K-Means algorithm to cluster the averages of daily event counts of the different APs in the training set into K components, with K ranging between 2 and 20. The centers of the K clusters were used as the parameters of the Λ exponential components, whereas the percentages of APs assigned to the different components of the mixture were used as weights W. Figure 1. shows how the log-likelihood changes with increasing number of components of the discrete mixture, for the training and test sets. Log-likelihood increases quickly for smaller numbers of clusters, and seems not to increase as fast for larger numbers of clusters; the standard deviation of the log-likelihood for larger number of clusters is larger than for smaller number of clusters.

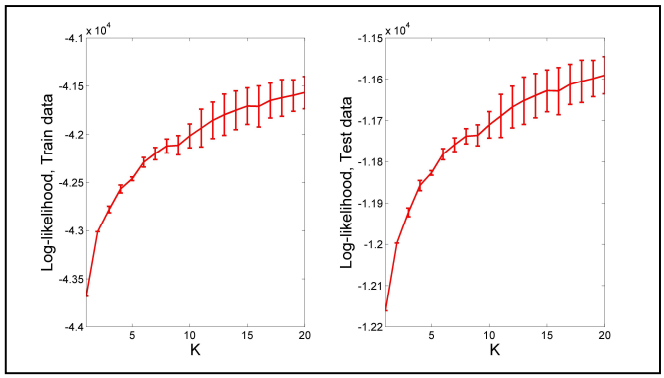

Figure 1. Log-likelihood of K-component exponential mixtures for the training set (left) and test set (right), and their 1-standard deviation error bars for 100 repetitions on each point. Notice that values for K=1 are equal to the results for the exponential model.

C. *Mixture of Exponentials with Gamma scale*

Although log-likelihood doesn't grow as fast for larger numbers of clusters, it does seem that larger numbers of clusters would provide better fitting to both training and test data. However, including more components in the discrete mixture increases the complexity of the model in the number of parameters: for each additional component of the mixture we need an additional λ_k and w_k. If we use a continuous mixture of exponentials with a parametric mixture model, we could get close to the best of both worlds.

We chose the Gamma distribution $\Gamma(\lambda; \alpha, \beta) = \beta^{\alpha}/\Gamma(\alpha).\lambda^{\alpha-1}\exp(-\beta\lambda)$ as the mixing model (also called scale) on the different values of λ. This is both because it is a conjugate prior of the exponential and because of the shape of the histogram of AP rates. The mixture distribution is then:

$$p_{mix\_exp}(x, \lambda; \alpha, \beta) = \Gamma(\lambda; \alpha, \beta)\lambda \exp(-\lambda x)$$

$$p_{mix\_exp}(x; \alpha, \beta) = \int_0^{+\infty} \Gamma(\lambda; \alpha, \beta)\lambda_a \exp(-\lambda x)\, d\lambda$$

$$= \alpha \frac{\beta^{\alpha}}{(\beta + x)^{\alpha+1}}$$

$$LL = N \ln \alpha + \alpha N \ln \beta - (\alpha + 1) \sum_{i=1}^{N} \ln(\beta + x_i)$$

The second form of p_{mix_exp} is the marginal distribution on x. For generating a sample with this model we first generate a λ from a Gamma distribution with parameters α and β, and then generate the sample from an exponential distribution with parameter λ.

Rather than fitting the distribution directly from the daily event counts, we fitted the Gamma distribution on the per-AP daily event count rate (i.e. one over the average). This gives us α and β for p_{mix_exp}. We obtained the following results on the training and test sets: $\alpha = 0.46686$, $\beta = 9.5469$, $LL_{train} = -4.2376E4$ and $LL_{test} = -1.1814E4$.

From Figure 1. we see that the continuous mixture model outperforms K-Means discrete mixture models with K <= 5. This is an interesting result as the continuous mixture model has only 2 parameters whereas the simplest discrete one that outperforms it has 9 (K=5, 2*K-1).

D. Above-Below AP Daily Event Count Average

In the case where we want to think about AP usage as high or low it may be useful to consider a binary variable θ that encodes whether a sample is above or below the average daily event count of its AP. We use a three-level model for generating samples. 1) For each daily event count sample we generate a value θ from a Gaussian distribution with support [0,1] and parameters μ and σ. 2) We use θ as the parameter of a mixture of two components, each of which is a Gamma mixture of exponentials, with parameters α_1, β_1 for the component above the AP average and α_0, β_0 for the component below. 3) We finally generate a sample daily event count from an exponential distribution λ which we draw from the mixture of two components. This yields the following joint PDF, where $\Theta = (\mu, \sigma, \alpha_1, \beta_1, \alpha_0, \beta_0)$ is the parameter vector and $\mathcal{N}_{[0,1]}(\theta; \mu, \sigma)$ is a Gaussian distribution with support [0,1].

$$p_{\text{abv blw}}(x, \lambda, \theta; \Theta) = \mathcal{N}_{[0,1]}(\theta; \mu, \sigma) \cdot \\ \cdot \{ \theta\, \Gamma(\lambda; \alpha_1, \beta_1)\, \lambda \exp(-\lambda x) + \\ + (1-\theta)\, \Gamma(\lambda; \alpha_0, \beta_0)\, \lambda \exp(-\lambda x) \}$$

Where

$$\mathcal{N}_{[0,1]}(\theta; \mu, \sigma) = \frac{\mathcal{N}(\theta; \mu, \sigma)}{\int_0^1 \mathcal{N}(\vartheta; \mu, \sigma) d\vartheta}, \theta \in [0,1]$$

The marginal distribution on x can be obtained by integrating the joint distribution over λ and θ. We first integrate on θ by computing $D(\mu, \sigma) = \int_0^1 \theta \mathcal{N}_{[0,1]}(\theta; \mu, \sigma) d\theta$ – the expected value of θ on $\mathcal{N}_{[0,1]}(\theta; \mu, \sigma)$ – using the standard error function erf (x). Then we integrate on λ using the result of the marginal distribution for the Gamma mixture of exponentials in the previous section. The marginal distribution on x is:

$$p_{\text{abv blw}}(x; \Theta) = D \frac{\alpha_1 {\beta_1}^{\alpha_1}}{(x + \beta_1)^{\alpha_1 + 1}} + (1 - D) \frac{\alpha_0 {\beta_0}^{\alpha_0}}{(x + \beta_0)^{\alpha_0 + 1}}$$

With

$$D(\mu, \sigma) = -\frac{\sigma}{\sqrt{2\pi}} \left(\exp\left(-\frac{(1-\mu)^2}{2\sigma^2} \right) - \exp\left(-\frac{\mu^2}{2\sigma^2} \right) \right) + \\ + \frac{\mu}{2} \left(\text{erf}\left(\frac{1-\mu}{\sqrt{2}\sigma} \right) + \text{erf}\left(\frac{\mu}{\sqrt{2}\sigma} \right) \right)$$

The log-likelihood of this model on sample set $X = (x_1, x_2, \dots, x_N)$ is:

$$LL = \sum_{i=1}^{N} \ln\left(p_{\text{abv blw}}(x_i; \Theta) \right)$$

For parameter fitting and for computing the log-likelihood on the training and test data, we first split the training data into above-below AP average sets. For estimating $\mathcal{N}_{[0,1]}(\theta; \mu, \sigma)$ we calculate θ_a, i.e. the percentage of daily event counts above AP average for each AP, and fit a Gaussian distribution on the resulting set of θ_a to get an estimate of μ and σ. Similarly, we calculate the daily event count rates for samples above and below the AP average, λ_a^{abv} and λ_a^{blw}, and fit two Gamma distributions resulting in estimates for α_1, β_1 and α_0, β_0 respectively. We obtained the following parameter estimates on our training set: $\alpha_1 = 0.92572$, $\beta_1 = 8.1737E1$, $\alpha_0 = 0.23254$, $\beta_0 = 0.22538$, $\mu = 0.41563$, $\sigma = 0.10866$. The log-likelihoods on the training and test sets using these parameters are: $LL_{train} = -3.9391E4$ and $LL_{test} = -1.1067E4$.

From Figure 1. we see that our above-below model outperforms K-Means discrete mixture models with K <= 20; in fact, this is true for all values of K that we tested. It also outperforms the exponential and Gamma mixture models. Although we increased the complexity of the model by including 6 parameters rather than 39 (K-Means with K=20), 2 (Gamma mixture of exponentials) and 1 (exponential), this is still remarkable.

VI. Considering Time – Binary Conditional Probability Models

A. Binary Conditional Probability Models

Considering time dependencies increases the complexity of models. However, if we can represent data with e.g. binary variables, we can use very simple conditional probability distribution (CPD) table models. Entries in these tables give us the probability of observing a value (e.g. the value to predict t_T) given a set of other values $(t_1, t_2, \dots, t_{T-1})$. Because we are talking about binary variables, the number of entries in the table of a T-variable problem is 2^{T-1}, which is a manageable number if T is reasonably small.

Estimating a CPD table from the training data amounts to counting the number of occurrences of the 2^T different combination of values of variables $(t_1, t_2, \dots, t_{T-1}, t_T)$ and dividing by the sum of the occurrences of $(t_1, t_2, \dots t_{T-1}, 0)$ and $(t_1, t_2, \dots, t_{T-1}, 1)$.

Generating samples from this model implies defining the temporal order between variables $(t_1, t_2, \dots, t_{T-1}, t_T)$. In this paper we are interested in daily event counts, so our time unit is the day and $day(t_T)$ gives us the index of the day for which t_T occurs. An example of temporal order in these variables is $day(t_i) = day(t_{i-1}) - 1$ for all $i \in \{2, \dots, T\}$. We assume t_T occurs after all the other variables. Once all samples prior to t_T have been independently generated, we

can use these samples as a lookup index in the 2^{T-1} table for the Bernoulli probability of the next sample.

A question that arises naturally is which variables $(t_1, t_2, \dots, t_{T-1})$ best help predict t_T. In this section we look into a simple model in which $(t_1, t_2, \dots, t_{T-1})$ are the $T-1$ previous samples to t_T. We set T ranging from 2 to 12 and use a uniform prior to build 11 table CPDs for each AP. We then calculate average and standard deviation for each table entry. We calculate the log-likelihood of these models on the training and test sets by summing 1) the Bernoulli log-likelihoods of the $T-1$ first samples of each AP, setting parameter θ to the estimate of the average of $\mathcal{N}_{[0,1]}(\theta; \mu, \sigma)$ of the above-below model, with 2) the Bernoulli log-likelihoods of the remaining samples of each AP with the parameter obtained from our models; the value of the parameter changes with sample according to the previous samples and the table CPD. The Bernoulli log-likelihood of binary variable x and parameter θ is defined as:

$$\mathrm{LL} = \begin{cases} \ln\theta & : \mathrm{x} = 1 \\ \ln(1-\theta) & : \mathrm{x} = 0 \end{cases}$$

Figure 2. shows the log-likelihoods of the 11 different models for the binary above-below training and test sets. The log-likelihood increases consistently on the training set whereas on the test set it peaks at T = 8. This indicates that whereas increasing T provides a better learning of the training data, at some point this is no longer beneficial for modeling the test data.

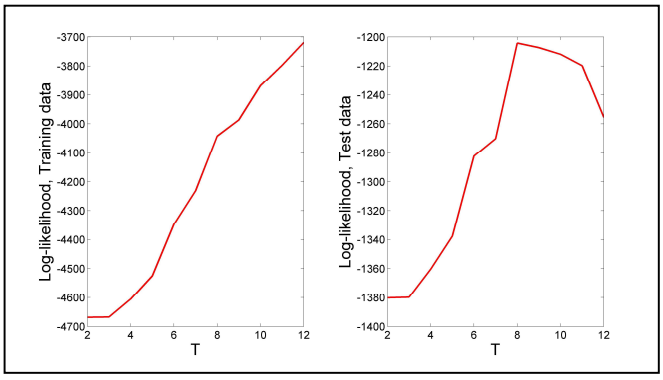

Figure 2. Log-likelihoods of the T=2:12 different models on the binary above-below training and test sets.

B. *Pluging-in Above-Below Average Models*

Notice that the log-likelihood values of Figure 2. cannot be compared directly with those of the previous sections because they are calculated on the above-below binary data sets, not the daily event count sets. To compare the time-binary models with the previous time-independent models that use daily event counts we use the above-below model as baseline and allow parameters μ and σ of the $\mathcal{N}_{[0,1]}(\theta; \mu, \sigma)$ distribution in the above-below model to change according to the CPD table values of the previous $T-1$ samples of the time-binary models.

To generate daily event counts using this mixed table CPD / above-below model, we start by generating $T-1$ samples for each AP using the above-below model. Then, we iteratively lookup the μ_{CPD} and σ_{CPD} parameters in the CPD table that correspond to the previous $T-1$ samples. We finally generate a daily event count sample from the above-below model with parameters μ_{CPD} and σ_{CPD}. We use the same log-likelihood function as the above-below model, except that now each sample can have a different set of μ and σ parameters.

Figure 3. shows the log-likelihoods of these models on the daily event count data sets. Notice that all these models have a log-likelihood above the previous models, which did not consider time. However, we must also consider that each of these models requires the 6 parameters of the above-below model plus $2 * 2^{T-1}$ parameters for the table CPD. Figure 3. seems to indicate that the $T = 6$ model with $6 + 2 * 2^5 = 38$ parameters is the best model for the test set amongst those presented previously. This is still better in terms of number of parameters than the K=20 K-Means model.

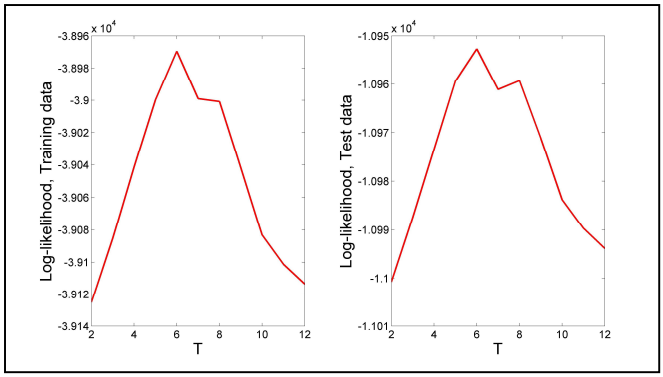

Figure 3. Log-likelihoods of the T=2:12 models on the daily event count training and test sets.

VII. CROSS-VALIDATION

In order to gain insight on the applicability of our model to different test and training data we use holdout cross-validation with the same number of training APs (143) and test APs (40) as the setup used in the rest of the paper. 100 different partitions of the data into training and test data were randomly generated and all the models presented in this paper were applied to these data partitions. TABLE I. presents the average and standard deviation of our models' parameters. Someone trying to generate synthetic data out of these results should sample a Gaussian distribution with the two right columns as parameters, for each model parameter they would like to use.

Log-likelihood results depend on the data the models are applied to. With cross-validation, data sets change from partition to partition and, as such, the log-likelihood results cannot be compared directly. We calculate average, standard deviation, minimum, and maximum differences between the log-likelihood of the base exponential model and the other models, on the same data set, for the 100 different training and test sets. These results are shown in Figure 4. and generally confirm our pre-cross-validation results: 1) the Gamma mixture model on average outperforms K<=5 K-Means discrete exponential mixture models although we did

get at least an instance on which it performed worse on the test data than the exponential model; 2) the above-below model always outperforms Gamma and K<=20 K-Means discrete mixture models within one standard deviation; and 3) binary T-1 previous samples model on average outperforms the other models, with a peak at T=6.

TABLE I. MODEL PARAMETERS

Parameter Name	**Average**	**St.Dev.**
# all-zero APs (training set)	2.3900	0.66507
# all-zero APs (test set)	0.61000	0.66507
θ_{zeros}	2.3379E-2	4.5867E-3
Exponential μ parameter	127.45	4.7637
Gamma mixture α parameter	0.52051	0.12000
Gamma mixture β parameter	12.646	7.7443
Above-below μ parameter	0.41566	4.5403E-3
Above-below σ parameter	0.10984	2.9368E-3
Above-below α_1 parameter	0.94505	0.10568
Above-below β_1 parameter	82.645	16.594
Above-below α_0 parameter	0.23709	5.7390E-2
Above-below β_0 parameter	0.30814	0.87809

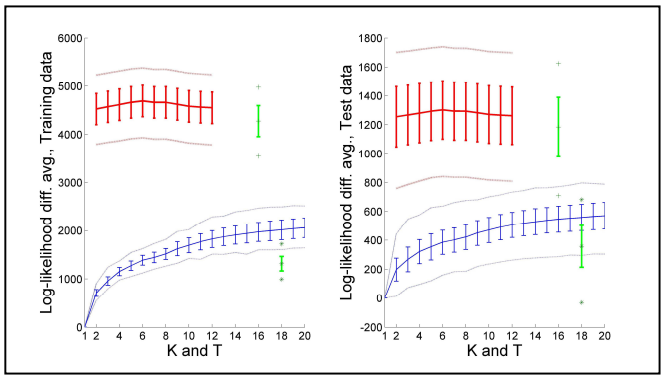

Figure 4. Log-likelihood differences from the base exponential model to 1) K-Means discrete exponential mixture with K=1:20 (K=1 is the baseline exponential); 2) Gamma mixture at T&K=18; 3) above-below model at T&K=16; and 4) table CPD at T=2:12. We show average and standard deviation (solid lines), as well as minimum and maximum (dotted lines) of these differences on 100 random hold-out cross-validation training (left) and test (right) data sets.

VIII. CONCLUSIONS

We presented an alternative approach to model 802.11 hotspots that focuses on AP usage and on daily keep-alive event counts proportional to the time users are connected to an AP, and used generative probabilistic models such as a Gamma mixture of exponentials and a binary above-below AP average model that makes it easier to consider dependencies between consecutive samples in time. We compared these models with the log-likelihood of the models on training and test data, a standard figure of merit in statistical learning. We conclude that the increase in complexity from the above-below model with 6 parameters to the T=6 table CPD model with 38 parameters (that has the best log-likelihood on both training and test sets) leads on average to a much smaller gain in log-likelihood than the increase in complexity from the Gamma distribution with 2 parameters to the above-below model with 6 parameters. These conclusions are supported by a 100-fold random holdout cross-validation.

The time-independent models that we presented do not consider commonalities within APs. In fact, a straightforward extension of using these models is to use the same parameters – such as λ, α, β depending on the distribution – when generating samples from the same AP. Although this is straightforward when generating samples, estimating maximum likelihood parameters and calculating the log-likelihood of these models can be much more complex. We also plan to further explore table CPD models and look into non-binary models for capturing time dependencies. We plan to apply our models to public available wireless data such as those from Dartmouth [2]. Finally, we plan to derive network traffic models from daily keep-alive counts to use in network simulations of different applications that run on wireless hotspot infrastructures such as the Eduroam campuses.

REFERENCES

[1] M. Afanasyev, T. Chen, G. M. Voelker, and A. C. Snoeren. Analysis of a Mixed-Use Urban WiFi Network: When Metropolitan becomes Neapolitan. Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement, 2008.

[2] Dartmouth's Crawdad website. http://crawdad.cs.dartmouth.edu/, last accessed March 2010.

[3] W. Hsu, D. Dutta, and A. Helmy. Mining Behavioral Groups in Large Wireless LANs. Proceedings of the 13th annual ACM international conference on Mobile computing and networking, 2007.

[4] M. Boc, A. Fladenmuller, and M. Dias de Amorim. Towards self-characterization of user mobility patterns. Mobile and Wireless Communications Summit, 2007.

[5] L. Song, D. Kotz, and R. Jain. Evaluating Next-Cell Predictors with Extensive Wi-Fi Mobility Data, IEEE Transactions on Mobile Computing, 2006.

[6] T. Henderson, D. Kotz, I. Abyzov. The changing usage of a mature campus-wide wireless network. In: Proceedings of the 10th annual international conference on mobile computing and networking (MobiCom), 2004.

[7] F. Hernandez-Campos, M. Papadopouli. A Comparative Measurement Study of the Workload of Wireless Access Points in Campus Networks. 16th Annual IEEE International Symposium on Personal Indoor and Mobile Radio Communications, 2005.