Using edge-reinforced random walks on graphs to model text

Peter Sunehag*
Statistical Machine Learning Group
NICTA/ANU Canberra, ACT 0200
Peter.Sunehag@nicta.com.au

Abstract

We introduce random walks on graphs with weighted vertices as a unifying framework for text models like LDA, Polya urns and a generalized Chinese Restaurant Process that we introduce here. We then suggest random walks on directed graphs with weighted edges as a way of going from exchangeability to partial exchangeability. This includes markov chain based models like LDA of markov chains. We will here suggest edge-reinforced random walks on graphs as a new model for text.

1 Introduction

We will here introduce the idea of using random walks on graphs as a unifying framework for many text models including LDA and Polya urns. We will begin by looking at undirected graphs with weighted vertices and introduce multinomials and Polya urns as random walks with or without reinforcement of the weights of the vertices we visit. We will continue by introducing a generalized Chinese Restaurant Process where the graph is constructed during our random walk. The generalized CRP will give us a probability distribution that is a normalization of an expression encountered by Madsen, Kauchak and Elkan in [7] as an approximation of a Polya urn. This gives us the possibility of performing perplexity experiments by plugging in the resulting parameters in the generalized CRP instead of in the Polya urn expression. The ultimate goal with the generalized CRP is to build a topic model based on it.
The second part of this project is concerned with random walks on directed graphs where we have put the weights on the edges instead of the vertices and given what vertex we are positioned at we randomly choose which edge to travel with probabilities proportional to the weights of the edges. A markov chain is then an unreinforced random walk where the weights are fixed and we will compare that to using an edge-reinforced random walk where the weight of a traversed edge and to a smaller extent the weight of the other edges leading to the same vertex is increased. In our experiments the edge-reinforced random walk has significantly lower perplexity than the markov chain. A key fact for the scalability of this algorithm is that a sequence of words can be translated into a sequence of bigrams in linear time. In the future a generalized CRP edge-reinforced random walk will also be considered.

*Long-term visitor at NICTA

2 Modelling text as Random walks on undirected graphs with weighted vertices

In recent years the Latent Dirichlet Allocation (LDA) model that was introduced by Blei, Jordan and Ng in [1], has been the focus of many papers. Given a corpus of documents, a number of topics are chosen and every document is modelled as a mixture of those topics. The topics themselves are multinomials. In [6], Madsen, Kauchak and Elkan suggest that Polya urn models are better suited for modelling text since it can capture the well known word burstiness property of natural languages. We will here view multinomial and Polya urn models as random walk on graphs models. Consider a complete undirected graph where every vertex has a loop. Attach positive weights to all vertices and draw the first vertex from a probability distribution where every vertex has a probability that is proportional to its weight. Then the random walk continues by giving all possible edges a probability that is proportional to the weight of the vertex on the other end of it. That is a multinomial model and if we adjust the model in such a way that we increase the weight of every vertex we step on by one, we have the Polya urn model. We will now introduce a new model that can be thought of as a generalization of the Chinese Restaurant Process. In this new model we will build up the graph while doing the random walk. We will start with just one vertex, the "add a new vertex" vertex, and we will attach weight s to it but no label corresponding to a word. Then, in the first step we put in a new vertex and we add an edge between the two vertices and we add a loop to the new vertex. Furthermore we put a weight of size one on this vertex and we draw a label (word) for this vertex from an urn of labels. After a label has been drawn nothing remains of that label in the urn. Labels can have different probabilities. In the next step we randomly choose an edge to traverse from the new vertex with probabilities proportional to the weight of the vertex on the other side. If we end up on "add a new vertex" we put in a new vertex, add edges between that vertex and all other vertices including itself and the "add a new vertex" vertex. We attach a weight of one to the vertex and we draw a label for it. After this we will continue from our new vertex. If we end up on an old labelled vertex we will increase its weight by one. When we write down our sequences of visited vertices we will only include labelled vertices. In this model it is necessary to either have at least as many labels as draws or adjust the model by adding a label "unknown 1" to the urn of labels and to have the rule that when "unknown 1" has been drawn, the label "unknown 2" is added to the urn in its place and with its probability. The probability of a specific sequence of n labels where we have visited exactly the $m \leq n$ different vertices with labels $k_1, ..., k_m$, appearing for the first time in that order and where the vertex with label k_j has been visited n_{k_j} times is

$$\frac{s^{m-1}}{(s-\beta_{k_1}) * ... * (s-(\beta_{k_1}+...+\beta_{k_m}))} \frac{\Gamma(s)}{\Gamma(s+n)} \prod_{j=1}^{m} \beta_{k_j}(n_{k_j}-1)! \tag{1}$$

where $\beta_w = sp_w$ and p_w is the initial probability of label w. The Polya urn p.d. gives probability $\frac{\Gamma(s)}{\Gamma(s+n)} \prod_w \frac{\Gamma(n_w+\alpha_w)}{\Gamma(\alpha_w)}$ to a random walk of length n where vertex w has been visited n_w times and where the vertices has weights α_w. In [7], Madsen, Kauchak and Elkan tried to get around the problem that it is very hard to maximize that expression. They used the approximation $\frac{\Gamma(x+\alpha)}{\Gamma(\alpha)} \approx \Gamma(x)\alpha$ for small α and they discovered that since the parameters that came out of performing MLE on a Polya urn on a corpus of documents are in general very small, much smaller than one, the parameters we get when we optimize the approximate expression are close to optimal for the original expression. The expression they maximized was $\frac{\Gamma(s)}{\Gamma(s+n)} \prod_{w:n_w \geq 1} \beta_w(n_w-1)!$. It is interesting here to point out that (1) is a normalization of that last expression. The first fraction in (1) is then the normalizing factor and in fact in our cases it is very close to one. It is possible for us to perform perplexity experiments by plugging in the parameters resulting from maximizing the approximating expression into (1) instead of in the Polya urn formula. The reason why

the generalized CRP is a close approximation to the corresponding Polya urn is that when we have very little weight on a vertex, say 0.001 or 0.002, then after that vertex has been visited by a vertex-reinforced random walk for the first time it will not matter very much if we had weight 0.001 or 0.002 since the difference afterwards is between 1.001 and 1.002. That is the idea behind the generalized CRP presented here. It can also be defined within the framework from [3] of adaptors and generators.

3 Modelling text as edge-reinforced random walks on directed graphs with weighted edges

In [4], Girolami and Kaban introduced Latent Dirichlet Allocation of Markov Chains which is a model that depends on transition counts. A probability distribution that only depends on k:th order transition counts is often called partially exchangeable of the k:th order or markov exchangeable of the k:th order. Just like an exchangeable p.d. can always be represented as a mixture of multinomials according to de Finnetti's representation theorem it is always possible to represent a partially exchangeable p.d. as a mixture of k:th order markov chains. In 1987, see [2], Diaconis and Coppersmith introduced edge-reinforced random walks on graphs which is a class of partially exchangeable models of the first order. We begin with an initial weight for every edge and given that our position is a certain vertex the probabilities for the different candidates for the next vertex to visit is proportional to the weights of the edges leading to them. If the weights are unchanged during the walk we have a markov chain but instead we will increase the weight of every edge that we traverse by one. That means that the probabilities of the edges that has already been used has been reinforced compared to those that has not yet been traversed. If we look at the probability distribution on word counts that is induced by an edge-reinforced random walk p.d. it will be a Polya urn just like a markov chain p.d. has multinomial word count probabilities. The philosophy presented here of how to go from "Bag of words" models to "Bag of bigrams" models is to define the new models such that they are refinements of the "Bag of words" models" in the sense they have the same kind of word count p.d.. Therefore we are particularly interested in sets of edge-weights with the property that if we sum all the weights of the incoming edges for a vertex, the sum should be equal to the corresponding parameter of a word count Polya urn p.d. that models the word counts well. Given a graph with weight assignments w_e for the edges, the probability of a certain walk $(v_1, v_2, ..., v_n)$, where v_i is the i:th vertex that we visit, can be written as the probability of the first word times the probabilities of all the transitions. That is $p(v_1, v_2, ..., v_n) = p(v_1)p(v_2|v_1, H_1)...P(v_n|v_{n-1}, H_{n-1})$ where H_i represent the history of how many times we have traversed each edge which is the information we need to calculate the edge-weights at that time. A transition from vertex v_i to vertex v_{i+1}, given that we are at v_i and that the current weights are w_e, has probability $\frac{w_{v_i,v_{i+1}}}{\sum_v w_{v_i,v}} = (\frac{w_{v_i,v_{i+1}}}{\sum_e w_e})/(\frac{\sum_v w_{v_i,v}}{\sum_e w_e})$ where e represents any edge in the graph. When we multiply together all the transitions, taking the changes in the weights into account we will end up with a Polya urn probability of bigram counts in the numerator and a Polya urn probability of word counts in the denominator. Thus it follows that

$$\frac{p(v_1, v_2, ..., v_n)}{p(v_1)} = \frac{\frac{\Gamma(s)}{\Gamma(s+n)} \prod_e \frac{\Gamma(y_e+\beta_e)}{\Gamma(\beta_e)}}{\frac{\Gamma(s)}{\Gamma(s+n)} \prod_v \frac{\Gamma(x_v+\alpha_v)}{\Gamma(\alpha_v)}} \approx \frac{\prod_{y_e \geq 1}(y_e - 1)!\beta_e}{\prod_{x_v \geq 1}(x_v - 1)!\alpha_v} \tag{2}$$

where x_v is the number of times we have passed by vertex v, counting the vertex we start with but not counting the vertex we stop on, y_e is the number of times we have traversed edge e and $\alpha_v = \sum_{e=(v,\tilde{v})} \beta_e$. We have used that $\frac{\Gamma(x+a)}{\Gamma(a)} \approx a\Gamma(x)$ for small a and when we take the logarithm of the rightmost expression in (4) summed over all training documents and differentiate it we will have an expression that vanishes whenever $\beta_e =$

$\frac{\sum_d I(y_{d_e} \geq 1)}{\sum_d I(x_{d_v} \geq 1)} \alpha_v$ for all $e = (v, \tilde{v})$. If we use the approximate optimization procedure from [7] mentioned in the previous section, to maximize the numerator in the middle expression of (2), this condition will be automatically satisfied. That means that we have translated our documents from sequences of words to sequences of bigrams, something which can be done in linear time, and then calculated the β_w parameters by fitting a bigram count Polya urn. The resulting word count Polya urn has the right proportions between its parameters but the total weight is much larger than the amount we receive by fitting a word count Polya urn directly which means that the word burstiness property has been reduced. If we would instead take the parameters from the Polya urn that is directly optimized over the word count data as our α_v and then calculate the β_e from those, we would get an unreasonably high probability bursting of traversed edges. This has lead us to the idea of using that weight assignement but with another reinforcement scheme where only a fraction of the new weight is added to the traversed edge and the rest is spread out over the other incoming edges to the vertex we are arriving to. This is the model that we have performed experiments on. More precisely we increase the weight of the traversed edge by $(1 + \lambda)/(W + \lambda)$ and the weight of the other edges by $1/(W + \lambda)$.

4 Preliminary experiments

In all our experiments we have used a standard probability smoothing to assign positive probability to things unseen in the training data by starting the transition counts for all possible transitions, given the vocabulary of the dataset, at $\epsilon > 0$ instead of at 0. It turns out that with this simple smoothing techinique, a Markov chain gives us similar perplexity as a multinomial and higher than a Polya urn model, but the edge-reinforced random walk is much better than the Polya urn. With more advanced smoothing techiniques we can expect the markov chain to improve compared to the multinomial and Polya urn but the edge-reinforced random walk should improve as well and continue to be the best model among the four. We have performed the experiments on the psychreview dataset (distributed with the Topic Modelling Toolbox by Griffiths and Steywers [5]) divided into 80 percent training data and 20 percent test data in five different ways. The result was a perplexity of 835 ± 53 for the multinomial, 839 ± 81 for the markov chain, 695 ± 32 for the Polya urn model, 405 ± 41 with the edge-reinforced random walk with $\lambda = 0$ and 390 ± 38 with $\lambda = 10$. The results are mean plus-minus one std. If we allow the algorithm to see which transitions will occur in the test data and only do our probability smoothing to those transitions then the markov chain is clearly better than both the multinomial and the Polya urn and the edge-reinforced random walk is still clearly better than the markov chain.

References

[1] Blei, D.M., Ng A.Y. & Jordan, M.I. (2003) Latent Dirichlet Allocation, *Journal of Machine Learning Research* **3**:993-1022.

[2] Diaconis, P., (1987) Recent progress on de Finneti's notion of exchangeability, *Bayesian statistics* **3**:111-125. Oxford University Press, New York 1988

[3] Goldwater, S., Griffiths, T., Johnson M. (2005) Interpolating Between Types and Tokens by Estimating Power-Law Generators, *Will appear in proceedings of NIPS 2005*

[4] Girolami, M., & Kaban, A. (2005) Sequential Activity Profiling : Latent Dirichlet Allocation of Markov Chains, *Data Mining and Knowledge Discovery*, **10**:175-196.

[5] Griffiths, T., & Steyvers, M. (2004). Finding Scientific Topics, *Proceedings of the National Academy of Sciences*, **101** (suppl. 1), 5228-5235.

[6] Madsen, R.E., Kauchak, E. & Elkan, C. (2005) Modelling word burstiness using the Dirichlet distribution *Proceedings of ICML, 2005*

[7] Madsen, R.E., Kauchak, E. & Elkan, C. (2005) Approximating the Dirichlet Compound Multinomial Distribution *Submitted to NIPS, 2005*