Achieving Fluency and Coherency in Task-oriented Dialog

Rashmi Gangadharaiah, Balakrishnan (Murali) Narayanaswamy, Charles Elkan
Amazon AI Lab
{rgangad@,muralibn@,elkanc@}amazon.com

Abstract

We consider real world task-oriented dialog settings, where agents need to generate both fluent natural language responses and correct external actions like database queries and updates. We demonstrate that, when applied to customer support chat transcripts, Sequence to Sequence (Seq2Seq) models often generate short, incoherent and ungrammatical natural language responses that are dominated by words that occur with high frequency in the training data. These phenomena do not arise in synthetic datasets such as bAbI, where we show Seq2Seq models are nearly perfect. We develop techniques to learn embeddings that succinctly capture relevant information from the dialog history, and demonstrate that nearest neighbor based approaches in this learned neural embedding space generate more fluent responses. However, we see that these methods are not able to accurately predict when to execute an external action. We show how to combine nearest neighbor and Seq2Seq methods in a hybrid model, where nearest neighbor is used to generate fluent responses and Seq2Seq type models ensure dialog coherency and generate accurate external actions. We show that this approach is well suited for customer support scenarios, where agents' responses are typically script-driven, and correct external actions are critically important. The hybrid model on the customer support data achieves a 78% relative improvement in fluency scores, and a 130% improvement in accuracy of external calls.

1 Introduction

Large or open domain chatbots are now omni-present, reaching many people through services like Microsoft's Cortana, Google's Assistant, Amazon's Alexa or Apple's Siri [1]. Recent efforts have been more focussed towards basic chit-chat [2, 3, 4, 5, 6] that are non-goal oriented. Chit-chat here refers to the ability to generate fluent responses, that are reasonable in the context of the conversation. In contrast, in task or goal oriented dialog, the chatbot needs to extract relevant information from the user (e.g. preferences), provide relevant knowledge to her (e.g. prices and availability), and issue appropriate system calls (e.g. make a payment). Very few approaches have been applied to goal-oriented settings but evaluated on synthetic datasets, such as bAbI [7] or through Wizard Of Oz experiments.

Advances in training deep neural networks has demonstrated the potential to build chatbots with minimal expert domain knowledge. In particular, supervised approaches such as Seq2Seq learning [2], can perform end-to-end learning from expert trajectories or dialogs, removing the need for many of the independent modules in traditional dialog systems, such as, the natural language understanding component, the natural language generation component, the dialog policy and the state tracker. Seq2Seq models (Section 3.1.1) achieve almost perfect performance on such synthetic datasets, however, they generate short, non-fluent and incoherent responses when tested on a real world dataset.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Table 1: Generated dialog on customer support data. *masked* used for anonymization.

Turns	User's Utterance	Agent's Response	Generated Agent's Response
Turn 1	got charged for <masked> membership after trial i did not want to continue.	hello , my name is <PERSON> . i 'm here to help you today .	hello <PERSON> , my name is <PERSON> . i 'm here to help you
Turn 2	<SILENCE>	thank you for being a <masked> member .	thank , for you <masked> member
Turn 3	<SILENCE>	i 'm sorry to hear that you were charged with our membership . no worries i 'll do my best	<SILENCE>
Turn 4	thank you , i did not know that it would auto renew after the trial	you 're welcome	you 're most welcome
Turn 5	<SILENCE>	i will now cancel your membership	i you sorry you while and you will issued
Turn 6	<SILENCE>	api_call cancel_refund	api_call cancel_refund
Turn 7	thank you so much	i hope my actions helped you out today	you 're most welcome
Turn 8	<SILENCE>	you 're most welcome	is there anything else i can assist you with with
Turn 9	very appreciated	is there anything else i can help you with today ?	<SILENCE>
Turn 10	no i 'm good	thank you for contacting <masked>	thank you for contacting <masked>

Fluency and Coherency: An example dialog generated by our best Seq2Seq model, trained on real customer-agent chat transcripts is shown in Table 1. We say a response is *coherent* if it is a sensible or a logical response considering the dialog context. Note that a coherent response does not necessarily guarantee fluency or grammaticality. For example, a sensible, coherent response at turn 4 in Table 1 would be to accept the customer's expression of gratitude as opposed to saying *how can I help you today ?* Consider the generated response at turn 2, the response is coherent because the agent is thanking the customer for being a member which is a sensible response that could occur at the beginning of the conversation, even though the response is ungrammatical or not fluent. Ideally, we want responses to be both *coherent* as well as *fluent* to ensure that the intended information is both conveyed and understood.

Coherence is a dialog level abstraction, any measure of which should correlate with how appropriate a response is in the context of the entire dialog history, while fluency is a measure of grammatical correctness of agents' responses at an utterance level. Measuring coherency is difficult in general. In our task oriented problems, we use task success as a measure of coherency. However, for example, consider the response at turn 5, the generated response is both incoherent as well as inarticulate, and fails to convey or represent the right information. Measuring dialog level coherence is important, and something we will address in future work.

We see that, when applied to real dialog datasets, Seq2Seq based models perform well with salutations and external actions but perform poorly on intermediate responses, a phenomenon we see throughout the dataset. In particular, they generate responses that are short, incoherent and ungrammatical. For example, the average utterance length generated by Seq2Seq is 4.2 words, compared with ground truth lengths that average 11.6. This is in part because end to end methods in general and Seq2Seq based models in particular, require large amounts of data before they are able to generate even somewhat fluent textual responses. To mitigate some of these problems, we explore nearest neighbor-based approaches. We show that such an approach is well suited to customer support scenarios, since the agents' responses are typically script-driven.

2 Related Work

Our work is closely related to retrieval-based chatbots [8, 9, 10]. These approaches formulate queries by concatenating all the utterances in the dialog history to create a bag of words representation to retrieve a response from a known set of responses. These approaches evaluate precision@K, from a restricted list, but do not indicate how this restricted list is obtained in practice. In contrast, we use a learned fixed size vector representation as a succinct summary of the dialog history, borrowing ideas from Seq2Seq generation-based chatbots [2, 3, 4].

Multi-class classification-based approaches [7] also pick one response (or class) from a fixed set of responses. A big disadvantage with such an approach is generating negative examples for each class. These approaches use poor selection strategies such as random sampling from the dialogs in the training data. This approach performed poorly when tested on our dataset, with most of the predicted responses being salutations, since salutations occur much more often in real world dialogs than rare problems. As the number of possible responses can grow significantly over time due to the heavy tailed nature of customer problems, such an approach increases the complexity of the dialog systems. In contrast, we explore nearest neighbor approaches that use vector representations or embeddings

obtained from the Seq2Seq-based models, and show that the performance scales well with the number and variety of dialogs.

Since many of the previously proposed approaches to task oriented dialog have been evaluated on synthetic datasets, it has been unclear how these approaches perform on real world tasks. In this paper, we propose an approach to generate both fluent and coherent responses, and test the approach on an internal customer support dataset.

3 Proposed Approach

This section describes our nearest neighbor approach for response selection. We see that such an approach is not able to accurately predict when to execute an external action. We show how to combine the nearest neighbor approach and Seq2Seq methods in a hybrid model, where nearest neighbor is used to produce fluent responses and Seq2Seq type models ensure dialog coherency and generate accurate external actions. We will explain each in some detail, with experiments showing the benefits of each and all together in combination. First we describe the datasets and metrics we use in our evaluations.

3.1 Dataset and Metrics

We first use data from the bAbI Tasks (Task1 and Task2) to evaluate our models. The other dialog tasks in bAbI require the model to mimic the knowledge base i.e., learn the entries in the knowledge base, making the knowledge base less useful once the model is trained. This is not a suitable strategy for our application, since knowledge bases undergo very frequent changes. In the bAbI task, the user interacts with an agent, in a simulated restaurant reservation application, by providing her constraints, such as place, cuisine, number of people or price range. The agent or chatbot performs external actions or SQL-like queries (api_call) to retrieve information from the knowledge base of restaurants, and make reservations. We refer to an entire session of text exchanges between an agent and a customer as a *dialog* and a *turn* refers to one interaction or a pair of text exchanges between the agent and the customer. Note that a system built using hand made rules can achieve 100% accuracy on this dataset, since it is synthetically created using a rule based generator. However, the goal of this paper and that dataset is not just to measure absolute performance, but to identify shortcomings of Seq2Seq-based models in a task oriented setting with no domain specific information. In our experiments, 80% of the data was used for training (of which 10% was used for validation) and the remaining 20% for testing the models. All the models were evaluated on the same test set.

In addition to the bAbI dataset, we also evaluate our models on an internal customer support dataset. An example dialog from this dataset is shown in Table 1. A customer support agent in this application, receives issues that belong to one of 31 possible categories. We consider one category, a subset of account issues, where solutions provided by agents were limited. We randomly sampled 1000 chat transcripts where the dialogs were not escalated, and also limited the number of turns to 20. Ideally we would want the model to also respond when user deviates into tangential topics, however, this would require the chatbot to know and understand general topics, but that is not the focus here.

The vocabulary size, maximum number of turns per dialog and the number of words per turn are much bigger than the bAbI dataset, making the task challenging. In addition, these transcripts are very noisy, with typographical errors. We perform spell correction, de-identification to remove customer sensitive information (such as names and addresses), lexical normalization particularly of lingo words such as *lol* and *ty*. Generalizing such entities reduces the amount of training data required. The values can be reinserted [11, 12] into the generated response. To preserve anonymity, we mask product and the organization name in the examples. One caveat to our evaluations is that they are based on customer responses to the actual human agent interactions, which are not fully indicative of how customers would react to the real automated system in practice. This is the difficulty of off-policy evaluation and learning. In ongoing work, we evaluate the system with real human agents providing utterance level scores for every dialog.

The fluency of dialog system outputs is typically evaluated using BLEU (BiLingual Evaluation Understudy [13]), which has been shown to highly correlate with human evaluation, BLEU is piecewise constant, and so models are usually trained to optimize categorical cross entropy loss [14, 15]. An advantage of high BLEU scores is that they indicate that the chatbot would produce

Table 2: Results with variants of the Seq2Seq model on the bAbI dataset.

Model	Type	Description	BLEU	P	R	Acc	EQM
Model 1	Basic Seq2Seq	dependencies between turns absent	88.3	0.60	1.00	0.87	0.00
Model 2	HRED Seq2Seq	Model1 + append $h_{L,enc}^{t-1}$	90.2	1.00	1.00	1.00	0.06

similar utterances in similar situations - reducing the problem of off-policy evaluation mentioned above. It should be noted that computing BLEU at turn level is also impacted by coherency since the generated response is compared against the response provided by a human agent. The use of evaluation metrics like BLEU or METEOR [16], to evaluate dialogs with just one reference has been debated [15]. There is still no good alternative to evaluate dialog systems, and so we continue to measure fluency using BLEU here.

Coherency also requires measuring correctness of the external actions which we measure using a metric we call Exact Query Match (EQM), which represents the fraction of times the api_call matched the ground truth query and we do not assign credits to partial matches. In addition, we report the precision (P), recall (R) and accuracy (Acc) achieved by the models in predicting whether to make an api_call (positive) or not (negative). These metrics help measure the timing accuracy of the api calls.

3.1.1 Seq2Seq Model

The basic Seq2Seq model [17] consists of two components, an encoder and a decoder, typically modeled using Long Short Term Memory Units (LSTMs) [18]. The encoder encodes the input sequence into a fixed dimensional vector, the output of the last hidden layer of the encoder. This vector is input to the decoder, which generates the output. In customer support, the input sequence is the user's utterance and the output is the agent's response. We use a novel variant of the hierarchical recurrent encoder-decoder (HRED) [5], to extend the basic Seq2Seq to handle multi-turn dialog. We unroll the basic Seq2Seq model, and make one copy for each turn present in the dialog. In Figure 1, orange-solid-square boxes represent the embedding and LSTM cells of the encoder. Green-dashed-square cells in the decoder represent the LSTM and dense layers with softmax for predicting each word in the agent's utterance or query sequence. The block arrows represent flow of information from one cell to another and arrows forking represent copies of this information. Each of the Seq2Seq copies share the same parameters. Once the training is complete, we use only one copy of the Seq2Seq model to make predictions. In order to make predictions for agent's responses at turn *t*, the context vector at turn *t-1* is appended to the embeddings of the user's utterance in turn *t*.

Table 2 shows results obtained with the vanilla Seq2Seq model which does not handle dialog history or context and results obtained with HRED. It can be seen that adding dependencies between turns substantially improved the perfomance across all the measures.

An improvement was also seen on the internal customer support dataset with HRED (Model 2 in Table 3). However, the generated responses were most often incoherent and not fluent. Table 1 shows an example response generated by a HRED variant, Model 2. Comparing the generated agent's response (column 4) with the true response (columns 3), it is evident that the model performs poorly on intermediate responses. This is because the outputs of Seq2Seq are dominated by words that occur with high frequency in the training data. As an example, you occurs with probability > 0.1 in the output of Seq2Seq, which occurs about one tenth as frequently in the training data. At a bigram level, much of the probability mass is assigned to ungrammatical combinations of frequent unigrams like *i* you.

We now proceed to explain the nearest neighbor based approach to produce reasonable responses that are more fluent.

3.2 Nearest Neighbor-based approach

In our nearest neighbor based approach to generating utterances, an agent's response is chosen from human generated transcripts - the training data - ensuring fluency. However, this does not necessarily ensure that the responses are coherent. The nearest neighbor approach starts with a belief state vector or embedding obtained from the entire history of the dialog so far, to improve coherency.

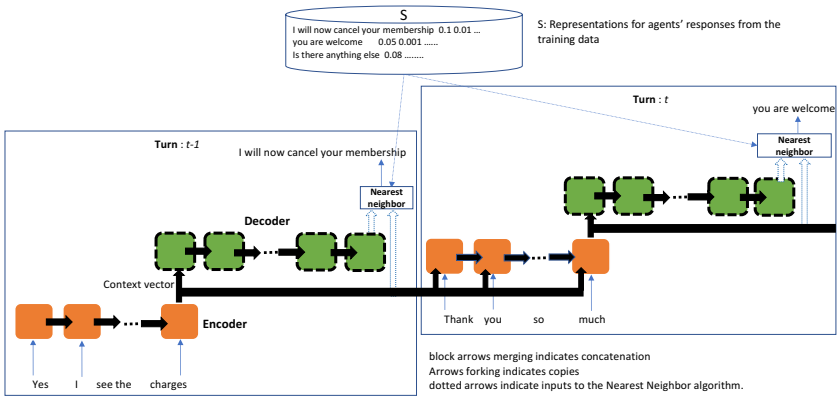

Figure 1: Proposed embeddings for finding the nearest neighbor.

Table 3: Results with the Nearest Neighbor approach on customer support data.

Model	Description	BLEU	P	R	Acc	EQM
Model 2	Seq2Seq	9.91	0.34	0.79	0.81	0.30
Model 3	Nearest neighbor using output of encoder	15.14	0.38	0.35	0.86	0.13
Model 4	Nearest neighbor using output of decoder	16.34	0.36	0.31	0.86	0.16
Model 5	Best Of both (Models 2+4)	17.67	0.33	0.73	0.80	0.30

Algorithm 1 Nearest Neighbor-based (NNB) approach

for $i = 1 : D$, ▷ D = number of test dialogs
  for $t = 1 : T$, ▷ T = max no. of turns in the test dialog$_i$
    if $t == 1$
      $testVec$ =BeliefStateRepresentation($user_t$) ▷ $user_t$ is the response of the user at turn t,
    else
      $testVec$ =BeliefStateRepresentation($user_{1:t}, agent_{1:t-1}$) ▷ $agent_p$ is the $true$ response of the agent at turn p

    $bs_{testVec}$ =NNB ($testVec$, S) ▷ find nearest neighbor of $testVec$ in S
    return $a_{testVec}$, agent's response represented by $bs_{testVec}$

3.3 Nearest neighbor with Seq2Seq embeddings

We propose using the outputs of the hidden units in the decoder and the encoder as suitable representations of the belief states. The Seq2Seq model for handling multi-turn dialog is trained as before (Section 3.1.1). Once the parameters have been learned, we proceed to generate representations of the state before each agents' responses in the training data. Pseudo-code for the nearest neighbor approach is in Algorithm 1. As seen in Section 3.1.1, the output of the last hidden unit of the encoder or the decoder at turn t, capture the dialog history well. This results in a tuple $< bs_{t,i}, a_{t,i} >$, where, $bs_{t,i}$ represents the belief state at turn t for dialog example i and $a_{t,i}$ represents the action the agent took while in this state i.e., the natural language response or api_call query issued by the agent. This is done to obtain belief state representations for all actions or agents' responses in the training set. This results in a set S that contains pairwise relationships between belief states and agent's actions. An agent's responses are scripted, and often multiple belief states will have the same action.

We test the models as done in Section 3.1.1, except now we do not generate the agent's response directly from the decoder. Figure 1 summarizes this process. We use the output of the last hidden unit of the decoder, $testVec$, to find the nearest neighbor $bs_{testVec}$ in S. We return the nearest neighbor's corresponding response, $a_{testVec}$, as the predicted agent's response. We use ball trees [19] to perform efficient nearest neighbor search. We can also concatenate the output of the last hidden unit of the encoder to the decoder's output to represent the belief state vector, shown as dotted arrows in Figure 1. We experimented with all three combinations, (1) considering the encoder's output, (2) considering the decoder's output (3) concatenating both.

Table 4: Generated dialog on customer support data using the hybrid model. *masked* used for anonymization.

Turns	User's Utterance	Agent's Response	Generated Agent's Response
Turn 1	i want to cancel my <masked> membership , please refund me	hello <PERSON> , my name is <PERSON> . thank you for being a <masked> member .	hello , my name is <PERSON> . i 'm here to to help you today .
Turn 2	hi , i found a bill list that charged me <MONEY>	<SILENCE>	please allow me a minute to check this for you .
Turn 3	i did not order <masked> membership	<SILENCE>	thanks for the details .
Turn 4	can you help me	please do not worry , i 'll be completely helping you with this .	i 'm sorry if any inconvenience happened to you .
Turn 5	thank you very much	you 're welcome .	i have successfully issued the refund for you and i will make sure this does not happen in future again .
Turn 6	<SILENCE>	<masked> successfully canceled , <MONEY> refund processes .	api_call cancel_refund
Turn 7	<SILENCE>	api_call cancel_refund	api_call cancel_refund
Turn 8	okay , good job	i 've already processes the refund in .	it was my pleasure assisting you today . thank you for contacting <masked> and do have a lovely time .
Turn 9	<SILENCE>	i 'm glad i could help you with this .	thank you for contacting <masked> , again this is <PERSON> and it 's my pleasure assisting you today .
Turn 10	<SILENCE>	is there anything else i can help you with ?	in the meantime , i want to make sure i have covered all of your concerns , please let me know
Turn 11	no more , thanks	it was my pleasure assisting a valued customer like you today .	please click on the end chat to close this window

Results obtained with this approach are in Table 3 (Models 3 and 4). Model 3 uses the output of the last hidden unit of the encoder only. Model 4 uses the output of the last hidden unit of the decoder. Both the models show a significant improvement in BLEU when compared to generating the agent's response directly from the Seq2Seq model (Model 2). Appending both the encoder's and the decoder's output did not show a significant change and hence, we do not report the results here.

The results also show that the Seq2Seq model achieved a better EQM when compared to the nearest neighbor approach (Models 3 and 4). The final hybrid model we propose, combines the best of both the strategies. We run both the Models 2 and 4 in parallel, when Model 2 predicts an API response, we use the output generated by Model 2 as the agent's response, otherwise we use the output of Model 4 as the predicted agent's response. This model achieved the best results among all models we study, both in terms of fluency (BLEU) as well as correctness of external actions (EQM). The hybrid model achieves a 78% relative improvement (from 9.91 to 17.67) in fluency scores, and 130% improvement in EQM over previous approaches (from 0.13 to 0.30).

4 Conclusions and Future Work

In this paper, we demonstrated limitations of Seq2Seq approaches, particularly on real world datasets. We proposed approaches, that combined the strengths of Seq2Seq models and nearest neighbor-based methods for task oriented dialog. We showed that this hybrid model was able to produce coherent and fluent responses. Table 4 shows an example response of the hybrid model. While many of the responses semantically match the true agent's response, they do not completely match the true response lexically. For example, *in the meantime, i want to make sure i have covered all your concerns, please let me know* matches *is there anything else i can help you with*, however, the choice of words to represent the intention is completely different. A disadvantage of using evaluation strategies such as BLEU, is that they unnecessarily penalize such valid responses. Although certain predicted responses do not match the true response semantically, they are still reasonable responses. Consider the predicted response, *i'm sorry for any inconvenience happened to you*, where the true response is, *please do not worry , i'll be completely helping you with this*, the responses convey different intentions but both are reasonable responses. The nearest neighbor approach produces redundant responses, for example, api_call *cancel_refund* is repeated in turn 6 and 7. As future work we will work on finding strategies to prevent such repetitive responses.

References

[1] Robert Dale. The return of the chatbots. *Natural Language Engineering*, 22(5):811–817, 2016.

[2] Oriol Vinyals and Quoc Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.

[3] L. Shang, Z. Lu, and H. Li. Neural responding machine for short text conversation. *arXiv preprint arXiv:1503.02364*, 2015.

[4] I. V. Serban, R. Lowe, L. Charlin, and J. Pineau. A survey of available corpora for building data-driven dialogue systems. *arXiv preprint arXiv:1512.05742*, 2015.

[5] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J. Nie, J. Gao, and B. Dolan. A neural network approach to context-sensitive generation of conversational responses. *NAACL*, 2015.

[6] J. Dodge, A. Gane, X. Zhang, A. Bordes, S. Chopra, A. Miller, A. Szlam, and J. Weston. Evaluating prerequisite qualities for learning end-to-end dialog systems. *ICLR*, 2016.

[7] Antoine Bordes, Y-Lan Boureau, and Jason Weston. Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683*, 2017.

[8] Zongcheng Ji, Zhengdong Lu, and Hang Li. An information retrieval approach to short text conversation. *arXiv preprint arXiv:1408.6988*, 2014.

[9] Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Dianhai Yu, Hao Tian, Xuan Liu, and Rui Yan. Multi-view response selection for human-computer conversation. In *EMNLP*, pages 372–381, 2016.

[10] Rui Yan, Yiping Song, and Hua Wu. Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, pages 55–64, New York, NY, USA, 2016. ACM.

[11] Jason D. Williams and Geoffrey Zweig. End-to-end lstm-based dialog control optimized with supervised and reinforcement learning. *CoRR*, 2016.

[12] Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, David Vandyke, and Steve Young. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562*, 2017.

[13] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic eval- uation of machine translation. *ACL*, 2002.

[14] R. Lowe, N. Pow, I. Serban, and J. Pineau. The ubuntu dialogue corpus: A large dataset for research in nstructured multi-turn dialogue systems. *arXiv preprint arXiv:1605.05414*, 2015.

[15] Chia-Wei Liu, Ryan Lowe, Iulian Vlad Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *CoRR*, 2016.

[16] Alon Lavie and Abhaya Agarwal. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. StatMT, 2007.

[17] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*, 2014.

[18] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[19] Ashraf M. Kibriya and Eibe Frank. An empirical comparison of exact nearest neighbour algorithms. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, PKDD 2007, pages 140–151, Berlin, Heidelberg, 2007. Springer-Verlag.