Predicting Cellular Link Failures to Improve User Experience on Smartphones

Alex Tom
alexgtom@stanford.edu

Srini Vasudevan
sriniva2@stanford.edu

Abstract—Cellular link failures cause poor user experience perceived as call drop or slowness/stall in data transfers. In this paper, we investigate a variety of supervised models that can predict link loss events. Based on this, the device could take proactive actions to avoid the link loss and improve the user experience in a seamless way. Features are chosen to represent wireless link performance and are averaged to mitigate temporal variations. We also introduce methods to improve precision while trading off with recall.

I. Introduction

End users desire great user experience on their smartphones—faster data access and less call drop. As more people adopt smartphones, it translates to more load on the cellular networks to carry additional traffic. The networks have over decades become heterogeneous, supporting different technologies such as 2G (GSM), 3G (UMTS), 4G (LTE) overlaid with each other. There are still occasions when the link fails due to variety of reasons—poor coverage in one technology coverage, high interference, poor network planning, lack of resources, etc. This link failure is perceived as either call drop during a voice call or slowness/stall in data transfer while browsing the internet. The link failure overall is an infrequent event (1-2% range), with some regions of higher percentages.

Our goal is to build a model in the device that can predict the link loss events. After the predicted link loss event the device could take preventive actions to avoid the link loss. Some examples of actions could be expedite handover to a new base station or different technology (LTE to UMTS or WiFi), increase transmission power, or prioritize voice over data. Such an algorithm will improve the user experience in a seamless way.

We use different supervised learning approaches. The training set is based on features learned from failures and non-failures conditions. The features chosen reflect the cellular link performance such as Receive power, noise estimate, transmit power, bytes transferred, error rates etc.

II. Features and Pre-processing

The features in our data set include different attributes of transmit (also referred as Uplink/Tx) and receive (Downlink/Rx) which indicate the nature of the wireless link. Some examples are receive power, receive signal to noise ratio, transmit power, link error rate, modulation scheme, amount of bytes transferred etc. These attributes are highly temporal. We take snapshots of these attributes and average them over time for feature extraction. The time window is chosen to be 5 seconds. Also to note that the attributes exhibit Gaussian distribution.

Our data only has two classes ($y = 1$ for link failure and $y = 0$ for no link failure) and are taken from processing of diagnostic information from cellular calls. Our data is skewed in that only 5% of our samples have label $y = 1$. We normalized the data and scaled all the features down so each feature has a mean of zero and unit variance. However in this application the class priors are imbalanced, this is due to the fact that link failures is a low probability event. This is problematic especially for generative models. To overcome this, the class samples were adjusted to balance the class priors.

III. Models

We applied LDA and QDA based on domain assumptions on the data set. Specifically that the data set was assumed to be a mixture of two multivariate gaussian distribution, since wireless link metrics are generally considered as gaussian random processes. After noticing lower f1 scores we wanted to explore other models for better prediction. We applied Logistics Regression to see if there is linear separability between the two classes. We also tried Naive Bayes as it is one of the simpler classifier to get a baseline. The f1 scores were still in the 60% range. This prompted us to explore non-parametric classifier, specifically K-neighbors which works well in non-separable spaces with multiple clusters. For K-Neighbors, we weighted the points by the inverse of their distance from the boundaries.

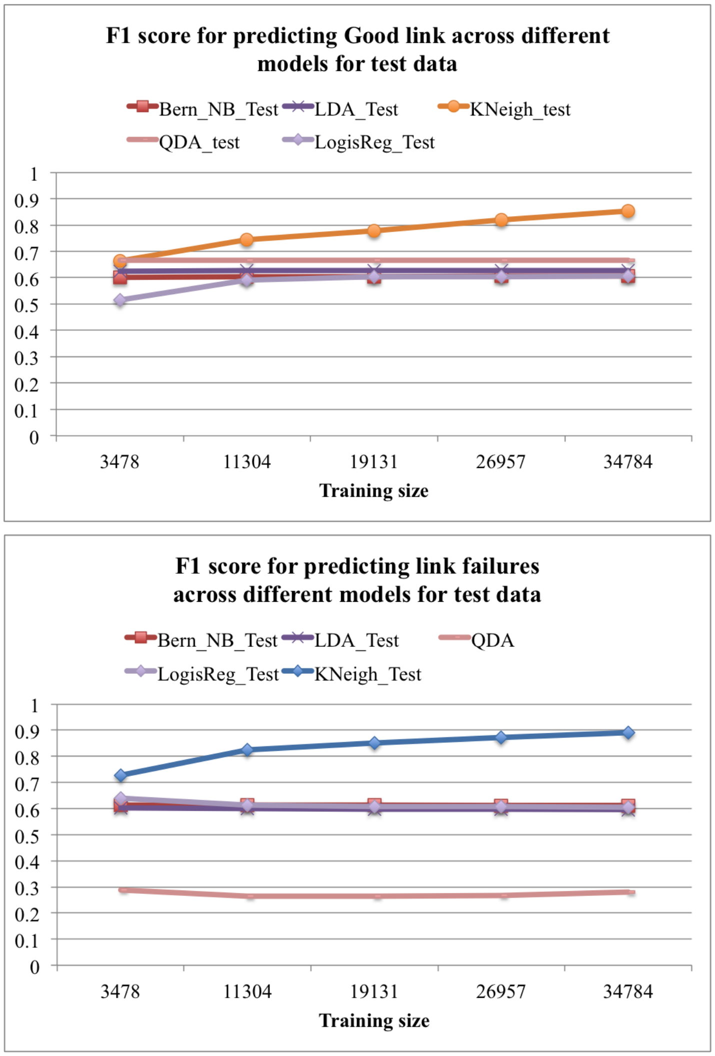

Figure 1: F1 scores for good links and link failures

	y = 0 (4422 samples)			y = 1 (4422 samples)		
	Precision	**Recall**	**f1 score**	**Precision**	**Recall**	**f1 score**
Logistic Regression	59.5%	62.0%	60.7%	61.7%	59.2%	60.4%
Naive Bayes	60.0%	61.2%	60.6%	61.8%	60.6%	61.2%
LDA	59.4%	66.2%	62.7%	63.3%	56.3%	59.6%
K-Neighbors	99.8%	74.6%	85.4%	80.3%	99.9%	89.0%
QDA	52.0%	92.5%	66.6%	71.0%	17.6%	28.2%

Table 1: Precision and Recall for each model from one of the k-folds

To evaluate the performance of each model, we used 5-fold cross validation where each fold had an even number of points from each label. We recorded the precision, recall, and f1 score of each model and picked the best model out of that. Our results are listed in Figure 1 and Table 1 above.

Notice that as the training size increases, the f1 scores for each model start to converge. It can be seen that K-Neighbors was best performing—99% precision with 90% F1 score in predicting link failure. This gives the intuition that there are "clusters" of points for link failure within good link conditions.

IV. Precision and Recall trade-off

Even though K-Neighbors gives good results, it requires access to entire training data set to make the predictions. This may not be a preferred approach in a smartphone for realistic implementation due to space and memory constraints. Hence, generative models will be more desired for this application. In this vein, we continue the rest of the discussion using LDA. Using LDA we get a f1 score of ~60% for $y = 1$ (predicting link failure).

	Y_{pred} = 1 (link failure)	Y_{pred} = 0 (good link)
Y = 1 (link failure)	TP	FN (misdetection/false alarm)
Y = 0 (good link)	FP (false alarm/misdetection)	TN

Table 2: Confusion matrix for y and $y_{predict}$

To understand the trade-off between precision and recall, lets us look at the above confusion matrix. While investigating from the point of view of predicting link failure class ($y = 1$), **FP** denotes the instances when the model predicted failure but the link turned out good. This represents the "false alarm" instances. In other words a vast number of devices will take wrong action which will be costly on the network and the end user (e.g. large users moving between two wireless technologies). Also note the percentage of failures in real world is very low compared to good cases (1:99). **Hence in this application it is extremely important to minimize FP.**

On the other hand **FN** denotes the instances when the model missed predicting the link failure. It is desirable to have FN low as well, however depending on the design goal we can trade-off one for the other. This means for this application, to improve precision we can reduce recall. Note: While investigating from the point of view of predicting good link class, FP will denote instances of misdetection and FN denotes false alarms. This shows the inter-dependency between the classes.

To achieve the trade-off, we used a threshold on the posterior scores from the model. The pseudocode is as below. The value THRESH is design choice on how much confidence is required to take action on the device.

```
if P(y = 0|X) > P(y = 1|X):
       predict_y = 0
else if P(y = 1|X) > THRESH:
       predict_y = 1
       proactive_action( )
else:
       predict_y = 0
```

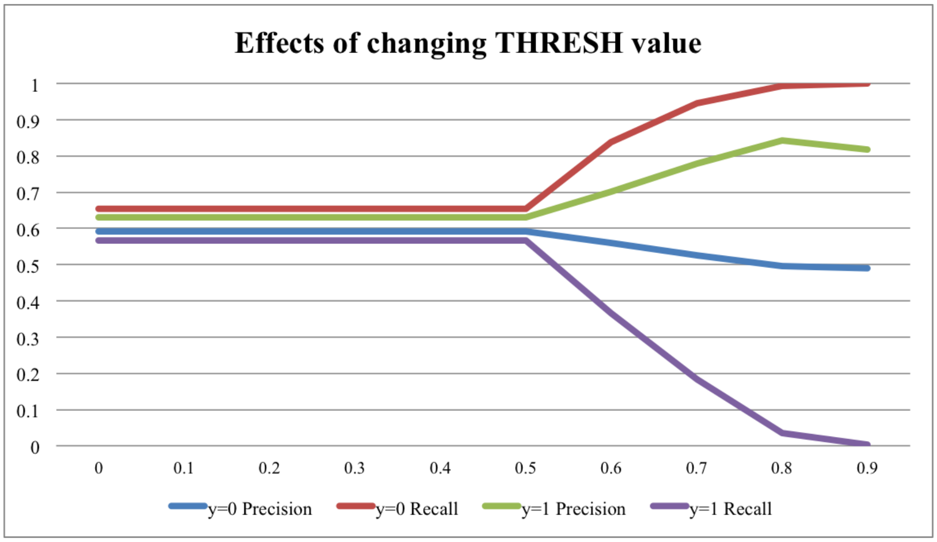

Figure 2: THRESH value of 0.7 provides a reasonable trade-off for link failure, ~80% likelihood of making the right decision and detecting ~20% of link fail cases.

V. Future: Subclassification

	Subclass 0	**Subclass 1**	**Subclass 2**
y = 0 (Training Set)	0.45%	99.54%	0.00%
y = 1 (Training Set)	93.72%	1.65%	4.62%
y = 0 (Testing Set)	0.56%	99.50%	0.00%
y = 1 (Testing Set)	91.91%	4.02%	5.39%

Table 2: Results for dividing the *y = 0* and *y = 1* into subclasses

We wanted to get further insight on the reason for failure i.e understand the underlying (hidden) reasons. We applied mixture of gaussian on each of the classes to determine the subclassification (signature), so that potentially different actions can be taken based on the signature. We divided the data further into three subclasses, however we need methods to make an interpretation of the different gaussian components and in choosing the right number of components using **B**ayesian **I**nformation **C**riteria scores.

VI. Conclusion

We used features from a cellular link to predict link failure, upon which pro-active measures could be used to improve user experience on smartphone. Non-parametric model such as K-Neighbors performed the best on the data set. However, for real time applicability we modified the decision on generalized model such as LDA by improving the precision and trading off with recall using a threshold parameter. This helped achieve 80% precision with 20% recall. With effective pro-active measures, smartphones can prevent 20% link failure and improve the overall user experience.

VII. References

Davis, Jesse, and Mark Goadrich. "The relationship between Precision-Recall and ROC curves." In *Proceedings of the 23rd international conference on Machine learning*, pp. 233-240. ACM, 2006.

Tsuruoka, Yoshimasa, and Jun'ichi Tsujii. "Boosting precision and recall of dictionary-based protein name recognition." In *Proceedings of the ACL 2003 workshop on Natural language processing in biomedicine-Volume 13*, pp. 41-48. Association for Computational Linguistics, 2003.