Cleaning Up Philly's Streets: A Cloud-Based Machine Learning Tool to Identify Illegal Trash Dumping

Abhinav Karale, William Kayat, Ameya Shiva,
Dr. Daniel Hopkins[*], Dr. Ani Nenkova[†]

Abstract

Philadelphia is currently struggling with short dumping, where citizens and businesses illegally dump waste and trash on the street, in vacant lots, etc. Short dumping costs the municipality around $8,124,375/year. Currently, police officers must manually monitor the over 450 CCTV cameras to identify acts of dumping. We are developing machine learning software which can be run on video extracted from city CCTV feeds to identify the timing and location of acts of illegal dumping as they happen. Our software will help the Streets department to monitor these dumping events much more systematically and will significantly speed up the manual work currently done by police officers. Our cloud-based solution enables us to train deep learning models on images of trash bags and then run them on video from street cameras to identify illegal dumps, saving the officers the time of going through the entire feeds.

Motivation and Value Proposition

Like many major cities, Philadelphia has to deal with lots of litter. One particular problem is short dumping, where citizens and businesses illegally dump waste and trash on the street, in vacant lots, etc. It costs the municipality $619 to clean up one ton of short dumps. In Dec. 2016, Philadelphias Mayor Kenney signed Order No. 13-16 – the Zero Waste Litter Cabinet – aiming to make Philadelphia waste-free by 2035. In late 2017, the city passed a bill to hike fines for short dumping, signalling their resolve to alleviate the issue. After meeting with a number of officials in the Streets and Sanitation departments over the past several months, we identified that most of the short dumping is due to commercial disposal of garbage and other products, particularly tires. From this helpful perspective, we are initially focusing on addressing the dumping problem as it relates to the specific items which are most frequently illegally disposed: black garbage bags and tires. From our discussions with the Deputy Commissioner in charge of sanitation, we realized that the majority of problematic dumping is caused by individuals who regularly and systematically dispose of trash on the street. For example, a mechanic normally has to pay a high price to legally dispose of tires, instead he can illegally dump the tires or pay someone to do so for him. The current solutions of fining $100 or raising awareness against dumping are not enough since it is still financially lucrative for them to short dump. The only way to challenge this behavior is to sanction the people who dump regularly, and the current resources deployed are simply not enough to consistently identify these offenders. We are developing machine learning software which, can be run on video extracted from city CCTV feeds, to identify the timing and location of acts of illegal dumping as they happen. Our software would enable the Streets department to monitor these dumping events much more systematically and therefore disincentivize the problematic behavior altogether in the long term. While interacting with the city to build our machine learning system, we uncovered some more fundamental problems with the short dumping reporting process. There was a lack of coordination between the people who watched the videos and identified short dumps and the members in the street department who had to ultimately record, respond, and clear up the dumps. The issue came down to both parties having different interfaces to do their work. The people watching the videos used internal tools to scan the videos and sent identified instances manually to the streets department, who ultimately were the ones who logged the information. Critical data was also lost/erroneous due to this process. For example, if the wrong timestamp or camera location was sent, the wrong decisions would be made and the wrong analysis would be performed. Thus, we realized that the city not only needed a more automatic solution to help flag short dumps, they also needed tools to close communication gaps between various stakeholders in the department. It was critical to build infrastructure not only on the machine learning side, but also on the software side so that city officials could all be on the same page about how to build standardized data systems to feed into the models in the future. We realized that a labelling interface could help not only acquire more data for the models, but also could be used to collaborate and share information about short dumps more easily across the department.

[*]Advisor (Department of Political Science)

[†]Advisor (Department of Computer Science)

Technical Approach

Our project utilized various technologies to accomplish the task at hand. We used Amazon S3 to store the video files, since the files were incredibly large (20+ GB) and we had to store them securely. We also used Amazon DynamoDB, a noSQL database, to store information about flagged instances in a video. We decided to use a noSQL database instead of a relational one because we wanted a flexible relational system instead of rigid relational tables as we experimented with user and data flows. To serve our API, we used Amazon Lambda in combination with API Gateway. Amazon Lambda allowed us to write serverless functions and deploy them seamlessly in the cloud. These functions had access to our S3 buckets and DynamoDB tables, so there was a powerful interface to interact with our data stores. Furthermore, API Gateway helped us build deployable endpoints that could interact with our front end systems. We could also use these endpoints in other backend systems if needed too, so having REST APIs that were easily callable and manageable made our work much easier. We also used EC2 instances during the machine learning portion of our project. Since our data environment had to be secured, we couldnt work on local installations of Python on our computer and access data (since it would then be stored in memory on the computer). Furthermore, we needed access to GPU resources to speed up training, so EC2 was ideal. We used the PyCharm IDE with an SSH interpreter to the EC2 instance, which allowed us to work without disrupting our code development workflow and still maintained our rigorous security standards. To make our work on the EC2 instance faster still, we would first copy selected files from S3 into the instance so access would be faster for training and testing. While this was an expensive operation upfront, it saved us a solid amount of time and was worth it.

Constructing the dataset of videos took some time. We needed to look at real security camera footage so the models we trained would be trained on the true data distribution and image quality. The streets department provided us with some of the real data. They provided us with data from seven different cameras in higher dumping areas and each video had about twenty-three hours of footage. We were also given some short clips of actual short dumps. However, the acquisition process was expensive. It took the city hours to first download the footage and then transfer it to our hard drive since the files were so large. The videos were stored in a legacy format. Since many Python image processing libraries work with .mp4s, we had to first convert the massive clips into the modern format and then upload it into our S3 bucket. It was difficult to keep asking for more clips since the process of even getting some data took a very long time.

There are two major technical components to our project. The first the collaborative platform and labelling interface, which involves both a front and back end. Our backend was using Flask and was written in Python. Our front end was built using React.js.

The backend has four main modules. The first one is an S3 module. This module has one GET endpoint to retrieve video information from S3 and DynamoDB. To access S3 videos on the front-end, we pre-sign the URLs (i.e. grant server-level permission for our front end client to access the S3 files). We grant the appropriate permissions in the backend to prevent exposing information on the front end. We also pull data from the DynamoDB instance. The key for each database entry is the video name, which is guaranteed to be unique. We also have a labelling module which has a single POST endpoint to update information about flagged instances for a given video. For example, if a new short dump is identified, we can use this endpoint to store that information in the database. It also allows for updating functionality. If someone mistakenly mislabels an example and then changes it, this module saves the update modification in the cloud. We also have a comment module with a single POST endpoint to add comments about a given video. Finally, we have a prediction module to actually run our model on the video. There is a single POST module that can handle two cases. The first is running the model on the entire video at a given sample rate (i.e. check for short dumps every 10 minutes). The second is running the model on a single image in the video for a point prediction.

The front end client serves as an interface for the back end. First, users can use a drop down menu to choose the video of interest. After loading that video, they can play it back in the browser itself. As they watch a video, if they identify a potential short dump, they can press a button to label that particular frame as a positive instance of a short dump. The button calls the labelling module internally to save that information. There is also a button to allow for a prediction on a single frame and another button that allows for a prediction on the entire video at the given sample rate. The crucial aspect we realized is that predictions can variable lengths of time. Short videos can be analyzed quickly, but longer ones can take minutes. Thus, we dont update the browser immediately after something is updated. The user has to refresh the page to see updated results. Users can always kick off predictions on a video and then check back later to see what is going on. When displaying the predictions and labels, we provide information on the timestamp of the particular video and what class it belongs too. We also include information if there is predictions available for that particular instance. There could be timestamps where the models label it differently than the human annotator does. More interestingly, it is also possible to have timestamps where the models have identified something but a human annotator has not looked at it yet. In that case, the human annotator can manually label it and change the label at any time.

On the machine learning side, we used PyTorch for our modeling. We framed the problem as image classification. Given an image, did it have trash bags in it? We tried a few different models. The first was one a simple pre-trained ResNet model. This model is trained on ImageNet, a large dataset of images for 1000 classes. There happened to be a class for trash bags called plastic_bag. This model outputted class probabilities for each of the 1000 classes, but we used the probability of plastic_bag and where that probability was in relation to all the other 999 classes to help explore potential decision rules. For example, if the probability of plastic_bag was in the top 10, then label the example as positive

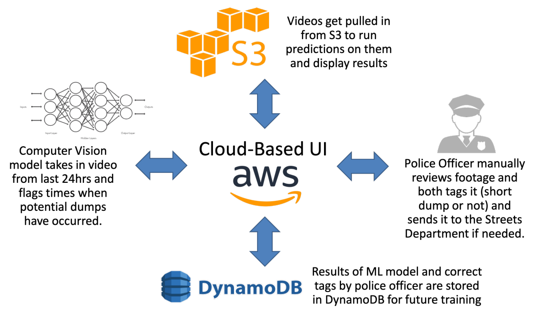

Videos get pulled in from S3 to run predictions on them and display results
S3
Computer Vision model takes in video from last 24hrs and flags times when potential dumps have occurred.
Cloud-Based UI
aws
Police Officer manually reviews footage and both tags it (short dump or not) and sends it to the Streets Department if needed.
DynamoDB
Results of ML model and correct tags by police officer are stored in DynamoDB for future training

Figure 1: Interaction between the different components of our project

(is trash). To experiment with the model, we took videos and sampled them to get images, which we then ran through the classifier. We also explored transfer learning. We knew that the pre-trained ResNet could be used to detect features of images, so we decided to look at the problem as binary classification. We removed the last layer of the network and added a new layer with only two units, one for positive and one for negative. We only trained this last layer. To get data for this approach, we scraped Google Images for 100-150 images of streets with trash in them and the same amount for streets without trash in them. We knew that since the network was already mostly trained, we only needed a small subset of examples to have a potentially decent model. Our third approach was using some data augmentation. We took about 100 images of streets and around 8-10 images of trash bags and we superimposed the trash bags on the pictures of the streets. We rotated, cropped, and sheared the trash bags. We also added Gaussian noise to both the trash bags and the street to make images more realistic and less similar. We also utilized some heuristicsfor example, trash bags should not be in the sky, so they should be in the bottom half of the image. We found ourselves constrained by the amount of positive data examples the city can give us. Deep learning models usually are data hungry, so we tried to get around that problem by leveraging pre-trained feature extractors in combination with other data tricks to avoid overfitting on the precious testing data we had. Since we have built a good deal of the machine learning infrastructure already, it is now much easier to update and re-train the models.

We learned a lot through this project. We learned how to use various AWS resources such as S3, DynamoDB, API Gateway, Lambda, and EC2 and how to make various services interact with each other. We also learned how to work securely in the cloud while still maintaining a smooth development process. We learned how to create backends in Python and deploy them to AWS, and how to write frontends in Javascript-based languages and deploy them to Heroku. On the machine learning side, we learned how to implement pre-trained models and how to create models that use transfer learning. We also learned how to scrape data effectively to create more high quality datasets. On a broader level, we realized how important having high quality data is and how an expensive data acquisition process leads to many model challenges that have to be taken into consideration.

Evaluation

On the interface side, we can mainly evaluate it qualitatively at this stage. We will presenting our software at the next enforcement subcommittee meeting at city hall on May 2nd, where we hope to get useful feedback from the potential users of our system. We quantitatively evaluated our ML components by their accuracy on the handful of test data real clips we received from the city.

- **Pre-trained model:** We first evaluated directly using the pre-trained resnet model on the data directly, to see whether the trash bag class was one of the most likely predicted classes. We tested the model out on three scenarios: 1) trash in a residential neighborhood during the day, 2) trash under a bridge at night, 3) a bridge at night without trash. The results were not great. On first, probability of trash can was .3% (40th overall class). On the second, probability of trash can was 1.1% (13th overall class). On the third, probability of trash can was .3%, (45th overall class). From these results, we realized that since the trash bags made up only a small portion of the images, we needed to come up with a threshold rule and add a classifier on-top of the pre-trained model to strengthen the models discriminative power.
- **Transfer learning**: The results from the pretrained model directly led us to convert the model to one of binary classification and retrain the layer of the network accordingly. With this approach, we were able to achieve an accuracy of 60.3% on the test image set. However, as our learning curves show in Figure 2, the model displayed clear signs of overfitting. This result made sense given we were working with a limited amount of data, and strengthened the case for collecting more and different data.
- **Augmented-Data**: After trying to create a synthetic train set using the techniques mentioned in the technical approach section, we found that our test accuracy was still around 60%, as we saw in the second approach. One of the issues we faced was that it was tough to generalize these synthetic images (where the trash was effectively pasted onto a scene) to the test set. Moreover, we had so little test data that we could not judge our accuracy results on the test set to be particularly indicative.

Overall, our evaluation of the various ML models and approaches we used suggested that a binary classifier was the

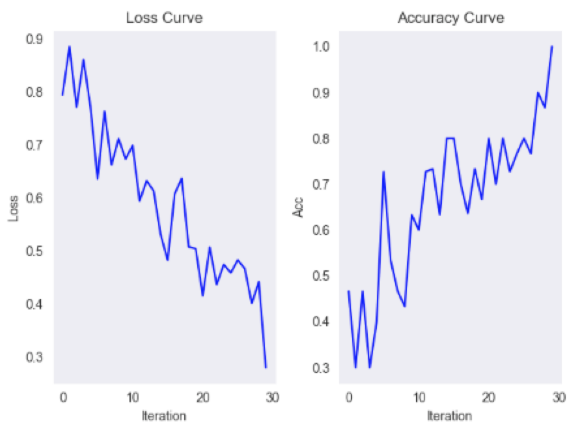


Figure 2: Learning Curves for Transfer Learning

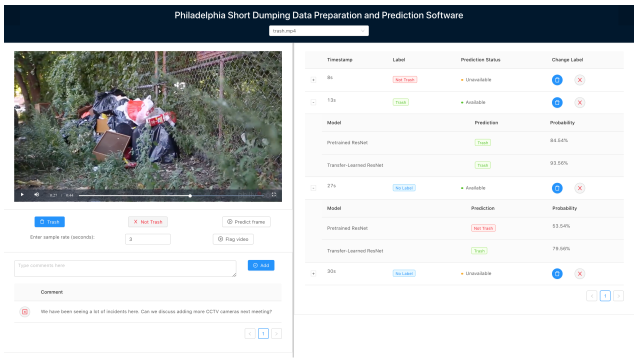
best approach. However, given our current limited data, particularly real-life positive examples to test on, we can not claim that our model works perfectly. We do believe that given more examples to train and test on, that the relevant metrics we consider here will improve significantly.

Discussion and Conclusions

Over the course of this project we found not only that our solution could considerably speed up the workflow of police officers trying to identify short dumps, but also help fix the broader problem of having to deal with enormous amounts of video data. The streets department is excited by the prospect of modernizing their technological infrastructure and having the tools necessary to monitor an increasing number of security cameras. Our project goes beyond addressing the streets departments needs by providing them with a fully integrated pipeline, to store, share and access the recordings taken from their cameras. Over the course of the year we found that the main issue is that police officers are overwhelmed with too much to data for a human to handle, and when they do happen to stumble upon the occurrence of a dump they must manually tell someone in the streets department to access the specified video and investigate the issue. Our cloud-based software provides both the streets department and the police with a centralized database of all videos, as well as past dumping instances. Our software can also run our deep learning classifier on any video on the fly, solving the original problem of automating the detection of illegal trash dumps.

Our major findings while working on this project were that the main need for automation the streets department has is with regards to commercial dumps, and they needed the technological infrastructure to run such analyses. This is what lead to the creation of our cloud software, which can run models that detect the presence of trash. Since the model can learn as more and more dumping instances are flagged and save, our solution continuously improves and does a better job at solving the citys need for automatic crime detection.

Although the accuracy of our classifiers is far from perfect, we are confident that we have setup all the right tools and put the infrastructure in place so that the development of fully finalized professional-grade software for the city is straightforward.


Philadelphia Short Dumping Data Preparation and Prediction Software

trash.mp4

Timestamp	Label	Prediction Status	Change Label
8s	Not Trash	Unavailable	
13s	Trash	Available	

Model	Prediction	Probability
Pretrained ResNet	Trash	84.54%
Transfer-Learned ResNet	Trash	93.56%

Timestamp	Label	Prediction Status	Change Label
27s	No Label	Available	

Model	Prediction	Probability
Pretrained ResNet	Not Trash	53.54%
Transfer-Learned ResNet	Trash	79.56%

Timestamp	Label	Prediction Status	Change Label
30s	No Label	Unavailable	

Enter sample rate (seconds): 3

Comment
We have been seeing a lot of incidents here. Can we discuss adding more CCTV cameras next meeting?

Figure 3: Interface for Police and Streets Department

Ethical Considerations and Societal Impact

This system could be used unethically by the city as an excuse to monitor certain individuals and areas (or infringe on their privacy) under the guise of trying to stop littering. However, we assume that the city is acting in the best interest of its citizens and our stakeholders are only concerned with identifying illegal trash dumping.

This project would only detect acts of dumping but not identify specific individuals doing so. It does however open the path to more advanced projects, for example monitoring CCTV cameras to detect violent crimes, drug deals, etc. This could pose some serious ethical issues, especially if the models involve identifying suspects and recognizing individuals. There are many questions that need to be answered before cities start monitoring their citizens with automated algorithms, especially including bias, how to deal with false positives, and many others.

Our current software includes a web-app allowing a human operator to review flagged dumping instances and annotate them as correct detections or false positives. In this way, we hope to improve our platform and also ensure that a human can supervise the software, to reduce issues of bias introduced by automating the detection process.

Business Analysis

Stakeholders

Philadelphia's Anti-Litter Task-force was only recently formed but has fairly influential members that want to make tangible change in the city. These are our primary stakeholders. They have tried steepening fines and cracking down harder, but these policies try to disincentive the behavior altogether rather than quickly or systematically identify when it has taken place.

Currently, security professionals within the police department have to constantly monitor CCTV feeds and use rules-of-thumb, well-known locations, previous experience, etc. to try to find short-dumping, and even if they do, they may be missing out on countless other instances. If they do spot a dump, they must physically call a counterpart in the Department of Streets and Sanitation and convey to them the relevant camera, date, and time to view the dump. Oftentimes, there is miscommunication through these legacy systems, leading to significant inefficiencies and wasted time. With our collaboration-supporting integrated infrastructure, we are streamlining the communication process between all parties focused on the issue of dumping by offering a central repository for the immediate flagging, labelling, and storing of clips and metadata. Beyond simplifying the workflow, type of system, they would be able to spot such activity more easily, and gain valuable insights about the circumstances surrounding such behavior.

Market Opportunity

Having a machine learning/AI system in place would help the city in two key ways:

1. Effectively identify situations involving illegally littering.
2. Give the city a framework that could be adapted for other related purposes (i.e. identifying other acts of littering, besides short-dumping, using a similar approach).

There are few, if any, competitors in this specific field. City-wide CCTV data is highly proprietary and is not accessible to the general public. Though we are starting with the narrow issue of trash-dumping, we believe getting cities and other camera networks on-board allows us to target a whole family of camera analytics tools that could be readily shared via our cloud-based infrastructure.

Customer Segment and Market Size

The video surveillance market has seen a significant rise in demand in recent year, due to negative incidents as well as the proliferation of more powerful cameras at a more economical price. 98 million network surveillance cameras are shipped globally through professional sales channels, and the global video surveillance system market was estimated to be $\approx$ \$35bn in 2017, forecasted to grow at a 15% 5-year CAGR. The software segment, and specifically, video analytics (currently $\approx$ \$5bn) is expected to grow at a faster pace than the broader market, at a CAGR of $\approx$ 20%. We believe our product is poised to capitalize on the growing demand for more comprehensive surveillance networks as well as the need to sift through the copious video data they generate to generate positive outcomes and financial savings for the end-users.

Competition

There are few, if any, competitors in this specific field. Surveillance-system manufacturers often include basic tools to view video, but very few provide an ML product to identify objects. City-wide CCTV data is highly proprietary and is not accessible to the general public. We therefore believe that our initial access to video data from the Philadelphia surveillance network would be a competitive edge for us, at least initially. Specifically, we are currently focused not on data acquisition, but analysis of the videos, which tends to have been largely ignored by many of the camera manufacturers who are focused on growing the total number of cameras on the streets, not necessarily on providing useful services to those monitoring the footage. The analytics software that they do provide is often focused on more obvious use object detection use cases (such as of license plates and facial surveillance). We think that access to these proprietary camera networks provides us an edge over some of the bigger cloud players with large AI/ML divisions, since the data distributions are unique. At the same time, our technology will help differentiate us from the camera makers, who have less expertise in this emerging field. We have yet to identify a competing software product on the market that can detect trash dumping or other simple yet high-impact issues.

The broader video surveillance market security has a number of large players, such as Bosch, Honeywell, Infinova, and several Chinese companies. Looking down the road, we think it could be advantageous to partner with a surveillance system manufacturer, which would help us connect with cities and camera networks already working with some of these players. Since our core offering is software, partnering would let us quickly scale our access points and deploy our solution.

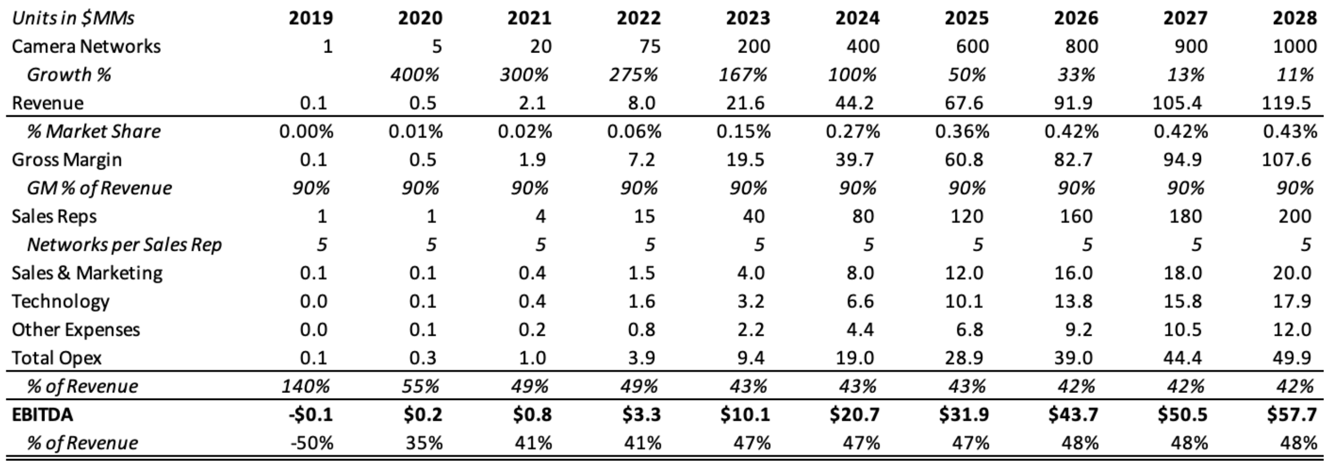
Financial Strategy

After examining a number of comparable companies providing services to the public sector, we are focused on developing a SaaS business model. Given that we are dealing with city governments and their camera networks, there is some upfront effort and cost required to acquire each city and hence customer. Additionally, sales-cycles can be longer when dealing with public agencies and the government. Once a customer is secured, though, we think our product would let us generate significant recurring revenues with a low amount of churn. Typical enterprise software packages across organizations can be around \$100K, which we use as our baseline view of the annual subscription price. We could also charge an additional fee based on the number of cameras our software covers (this is used by a private comp that we examined) in the network or for certain large volumes of video processed using our software. A 10-year revenue forecast is given in Table 1, assuming pricing grows at inflation. Implied market share within the security surveillance analytics market is given for reference.

On the expenses side, acquisition of customers (sales and marketing expense) would be one of the major costs. We anticipate hiring sales reps to help acquire and manage customers, and we have forecasted these costs accordingly (1. On the engineering side, our product is highly scalable (main costs are hosting overhead, etc). A detailed financial model has been included in Figure 4.

Acknowledgements

Our sincerest thanks to our advisors Dr. Ani Nenkova and Dr. Dan Hopkins for their support during this project. We also extend our thanks to Deputy Commissioner Richard Montanez, Nic Esposito and everyone at the Philadelphia Streets Department, as well as Dr. Christine Brisson for helping us set up the infrastructure for this project.



Units in $MMs	**2019**	**2020**	**2021**	**2022**	**2023**	**2024**	**2025**	**2026**	**2027**	**2028**
Camera Networks	1	5	20	75	200	400	600	800	900	1000
Growth %		*400%*	*300%*	*275%*	*167%*	*100%*	*50%*	*33%*	*13%*	*11%*
Revenue	0.1	0.5	2.1	8.0	21.6	44.2	67.6	91.9	105.4	119.5
% Market Share	0.00%	0.01%	0.02%	0.06%	0.15%	0.27%	0.36%	0.42%	0.42%	0.43%
Gross Margin	0.1	0.5	1.9	7.2	19.5	39.7	60.8	82.7	94.9	107.6
GM % of Revenue	*90%*	*90%*	*90%*	*90%*	*90%*	*90%*	*90%*	*90%*	*90%*	*90%*
Sales Reps	1	1	4	15	40	80	120	160	180	200
Networks per Sales Rep	*5*	*5*	*5*	*5*	*5*	*5*	*5*	*5*	*5*	*5*
Sales & Marketing	0.1	0.1	0.4	1.5	4.0	8.0	12.0	16.0	18.0	20.0
Technology	0.0	0.1	0.4	1.6	3.2	6.6	10.1	13.8	15.8	17.9
Other Expenses	0.0	0.1	0.2	0.8	2.2	4.4	6.8	9.2	10.5	12.0
Total Opex	0.1	0.3	1.0	3.9	9.4	19.0	28.9	39.0	44.4	49.9
% of Revenue	*140%*	*55%*	*49%*	*49%*	*43%*	*43%*	*43%*	*42%*	*42%*	*42%*
EBITDA	**-$0.1**	**$0.2**	**$0.8**	**$3.3**	**$10.1**	**$20.7**	**$31.9**	**$43.7**	**$50.5**	**$57.7**
% of Revenue	-50%	35%	41%	41%	47%	47%	47%	48%	48%	48%

Figure 4: 10-Year Financial Projections