DEEP LEARNING WITH MICROSOFT COGNITIVE TOOLKIT CNTK IN REAL LIFE PROJECTS

LIUBOV KAPUSTINA, DATA SCIENTIST
IGOR YAROSHENKO, DATA SCIENTIST

AI UKRAINE
13-14 October, 2018
Liubov Kapustina is a data scientist with more than 10 years’ experience in the industry. She has experience in risk scores and predictive modeling in Banks and in consulting. She has wide experience of realized real production projects with AI and Data science. She also participated in the Marketing Revolution 2015 and AI Ukraine 2015 conferences as a speaker. She is member and speaker in Kiev Big Data Community.

https://www.linkedin.com/in/liubov-k-03164525/
Igor Yaroshenko is young and ambitious data scientist with strong technical background. He has experience in bank industry and e-commerce projects. He has wide experience of realized real production projects with AI and Data science. Igor has participated in the conference Nordic Business Day 2016 as a speaker with talk “Data Science innovation in Agriculture (smart hive)” and Kyivstar Big Data Hackathon.

https://www.linkedin.com/in/igor-yaroshenko-0468b369/
Part I: how we choose CNTK + c#

- Access Softek company, our tasks, data volumes
- Comparison CNTK with other existing solutions
- 3 pillars of our requirements: speed, fast and cheap training and accuracy

Part II: CNTK + c# in real life projects

- Fraud detection
- User Engagement
- Churn Prevention
Part I: how we choose CNTK + c#

- Access Softek company, our tasks, data volumes
- Comparison CNTK with other existing solutions
- 3 pillars of our requirements: speed, fast and cheap training and accuracy

Part II: CNTK + c# in real life projects

- Fraud detection
- User Engagement
- Churn Prevention
mobile banking

Over 3 million active users per month

400+ financial institution clients

online banking
Part I: how we choose CNTK + c#

- Access Softek company, our tasks, data volumes
- 3 pillars of our requirements: speed, fast and cheap training and accuracy
- Comparison CNTK with other existing solutions

Part II: CNTK + c# in real life projects

- Fraud detection
- User Engagement
- Churn Prevention
Why are these requirements so important?

Firstly, the user must get permission or be prohibited from the transaction in real-time. For a backend serving requests from multiple clients, C# with just-in-time compilation is incomparably more efficient than Python.

Secondly, there are very few examples of fraud, and they are often unique. It’s like a flu virus: once we have taught the model to define one type, a new type can immediately appear.

How quickly should the model learn when new data is available?
As soon as possible!

Microsoft Cognitive Toolkit library trains LSTM-models up to 4 times faster than TensorFlow, it parallelizes the processes and uses the server’s resources more effectively and efficiently. In this way we can train the model faster and cheaper.

Under The Nilson Reporter, in 2017, for every $100 spent, 7.2 cents is lost due to fraud, therefore the fraud prevention cannot cost more than this, otherwise it will become more expensive than the fraud itself.

Thirdly, the accuracy of TensorFlow and CNTK backends are similar across all benchmarks, and in some of it CNTK is better.
Part I: how we choose CNTK + c#

- Access Softek company, our tasks, data volumes
- 3 pillars of our requirements: speed, fast and cheap training and accuracy
- Comparison CNTK with other existing solutions

Part II: CNTK + c# in real life projects

- Fraud detection
- User Engagement
- Churn Prevention
## SPEED, FAST AND CHEAP TRAINING AND ACCURACY

*The latest (16th) Techempower benchmarks*

<table>
<thead>
<tr>
<th>Performance</th>
<th>Responses per second</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single query: database-access</td>
<td></td>
</tr>
<tr>
<td>responses per second, single query</td>
<td>15,470</td>
</tr>
<tr>
<td>Multiple queries: Responses per second at 20</td>
<td></td>
</tr>
<tr>
<td>queries per request</td>
<td>420,820</td>
</tr>
<tr>
<td>Data updates: responses per second at 20 updates</td>
<td></td>
</tr>
<tr>
<td>per request</td>
<td>6,688</td>
</tr>
<tr>
<td>Plaintext responses per second</td>
<td>1,822,366</td>
</tr>
<tr>
<td>Fortunes, which is a test of read performance</td>
<td>45,230</td>
</tr>
</tbody>
</table>

C# could have a big draw near this respect. C# is enforced as compiled language. If we talk about performance, C# is faster due to its Common Language Infrastructure Framework.

Python is enforced as associate taken the language. With Python implementation of the JIT compiler, Python’s program execution is improved.
### SPEED, FAST AND CHEAP TRAINING AND ACCURACY

*The latest (16th) Techempower benchmarks*

<table>
<thead>
<tr>
<th>Performance</th>
<th>C# could have a big draw near this respect. C# is enforced as compiled language. If we talk about performance, C# is faster due to its Common Language Infrastructure Framework.</th>
<th>Python is enforced as associate taken the language. With Python implementation of the JIT compiler, Python program execution is improved.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single query: database-access responses per second, single query</td>
<td>15,470</td>
<td>4,398</td>
</tr>
<tr>
<td>Multiple queries: Responses per second at 20 queries per request,</td>
<td>185,435</td>
<td>420,820</td>
</tr>
<tr>
<td>Data updates: responses per second at 20 updates per request</td>
<td>6,688</td>
<td>4,390</td>
</tr>
<tr>
<td>Plaintext responses per second</td>
<td>1,822,366</td>
<td>975,225</td>
</tr>
<tr>
<td>Fortunes, which is a test of read performance</td>
<td>45,230</td>
<td>20,414</td>
</tr>
</tbody>
</table>
Part I: how we choose CNTK + c#

- Access Softek company, our tasks, data volumes
- 3 pillars of our requirements: speed, fast and cheap training and accuracy
- Comparison CNTK with other existing solutions

Part II: CNTK + c# in real life projects

- Fraud detection
- User Engagement
- Churn Prevention
SPEED, FAST AND CHEAP TRAINING AND ACCURACY

We report 8 GPUs (2 machines) for CNTK as it is the only public toolkit that can scale beyond a single machine. CNTK on Azure GPU Lab can scale beyond 8 GPUs across multiple machines with superior distributed system performance.

Theano only supports 1 GPU.
SPEED, FAST AND CHEAP TRAINING AND ACCURACY

CNTK IS MUCH FASTER!

Speed of Bidirectional LSTM Approach on IMDb Data

- CNTK
- TensorFlow

Average Epoch Runtime (seconds)
Part I: how we choose CNTK + c#

- Access Softek company, our tasks, data volumes
- 3 pillars of our requirements: speed, fast and cheap training and accuracy
- Comparison CNTK with other existing solutions

Part II: CNTK + c# in real life projects

- Fraud detection
- User Engagement
- Churn Prevention
SPEED, FAST AND CHEAP TRAINING AND ACCURACY

Performance of Bidirectional LSTM Approach on IMDb Data

Normally the accuracy *increases* as training proceeds; Bidirectional LSTMs take a long time to train to get improving results, but at the least both frameworks are equally performant.
SPEED, FAST AND CHEAP TRAINING AND ACCURACY

Performance of CNN Approach

Test Accuracy (Higher is Better)

Epoch

- CNTK
- TensorFlow
SPEED, FAST AND CHEAP TRAINING AND ACCURACY

Performance of Text Generation via LSTM

Loss (Lower is Better)

Epoch

- CNTK
- TensorFlow
Part I: how we choose CNTK + c#
- Access Softek company, our tasks, data volumes
- 3 pillars of our requirements: speed, fast and cheap training and accuracy
- Comparison CNTK with other existing solutions

Part II: CNTK + c# in real life projects
- Fraud detection
- User Engagement
- Churn Prevention
PART II: CNTK + C# IN REAL LIFE PROJECTS
Part I: how we choose CNTK + c#

- Access Softek company, our tasks, data volumes
- 3 pillars of our requirements: speed, fast and cheap training and accuracy
- Comparison CNTK with other existing solutions

Part II: CNTK + c# in real life projects

- Fraud detection
- User Engagement
- Churn Prevention
PART II: CNTK + C# IN REAL LIFE PROJECTS: FRAUD DETECTION

Overview
Detect, Predict, and Prevent Fraud, in real-time.

Fraud Control is designed to learn and react to each member's banking usage, to keep accounts safe and fraud under control.

• Detect suspicious behaviors
  • Fraud Control's eyes are unsupervised and semi-supervised machine learning techniques to cluster and classify out-of-pattern account-level behaviors

• Predict fraud risk
  • Fraud Control's brain is neural network ensemble that learns to score suspicious behavior based on admin feedback to fraud discovery and user performance on event-triggered layered authentication

• Prescribe preventative countermeasures
  • Fraud Control's hand is an API that integrates with the banking channel's layered authentication, to safeguard the app from fraudsters

• Discover new ways to control fraud
  • Fraud Control's face is a reporting and visualization tool to help admins break down user activity, global alerts, specific fraud threats, and layered authentication performance.
What does Fraud Control see?

Real-time User Behaviors
• Member ID
• Session ID
• Date & Time
• User Operation
• Screen / Feature / Operation Group
• App Response
• Server Response Duration
• Transaction Value (where applicable)
• Application Version
• Client
• Device IP Address
• Device GPS (if enabled by end-user)
• Device OS Version
• Device Model

What does Fraud Control find?

• Surprising location
• Using VPN
• New or unconventional device
• Large or suspicious money movement
• Activity at atypical time or date
• Unusual user behaviors
• Suspicious app responses
• Etc...

What does Fraud Control do?

• Constantly learns from every data point sent from all devices and servers accessing the banking app
• Detects and scores suspicious behavior on the account
• Integrates with Layered Authentication to control access to app functions
• Constantly learns from feedback to improve its predictions over time
PART II: CNTK + C# IN REAL LIFE PROJECTS: FRAUD DETECTION

Five Fraud Risk Components

1. User Behavior Risk
2. Location Risk
3. Money Movement Risk
4. Time Risk
5. Device Risk

Concept is follows:
1. Separate 2 types of factors Linear and Matrix (ordered by steps)
2. For Linear - analyses distribution and "typical"
3. For Matrix - probability to meet event "A" on step "N"
4. Analyze by Model difference from current to typical
5. Aggregate all metrics on Session level
6. Choice how calculate 5 sub-scores in final Fraud Risk Score
PART II: CNTK + C# IN REAL LIFE PROJECTS: FRAUD DETECTION

User Behavior Risk

Low fraud risk sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Finger Print Login</th>
<th>Login</th>
<th>Money Transfer</th>
<th>Screen balance</th>
<th>Screen Bill Pay</th>
<th>Money transfer</th>
<th>Screen balance</th>
<th>Logout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session 1</td>
<td>21</td>
<td>30</td>
<td>40</td>
<td>21</td>
<td>95</td>
<td>19</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>Session 2</td>
<td>16</td>
<td>27</td>
<td>42</td>
<td>20</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Session 3</td>
<td>18</td>
<td>32</td>
<td>89</td>
<td>20</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Session 4</td>
<td>21</td>
<td>25</td>
<td>37</td>
<td>93</td>
<td>22</td>
<td>10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Risk Score

- Session 1: 0.226
- Session 2: 0.114
- Session 3: 0.092
- Session 4: 0.007
PART II: CNTK + C# IN REAL LIFE PROJECTS: FRAUD DETECTION

User Behavior Risk

High fraud risk session

<table>
<thead>
<tr>
<th>Session</th>
<th>Finger Print Login</th>
<th>Login</th>
<th>Screen balance</th>
<th>Money Bill Pay</th>
<th>Screen Bill Pay</th>
<th>Screen balance</th>
<th>Money Transfer</th>
<th>Logout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session 1</td>
<td>21</td>
<td>30</td>
<td>40</td>
<td>21</td>
<td>95</td>
<td></td>
<td>19</td>
<td>12</td>
</tr>
<tr>
<td>Session 2</td>
<td>16</td>
<td>27</td>
<td>42</td>
<td>20</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Session 3</td>
<td>18</td>
<td>32</td>
<td>89</td>
<td>20</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Session 4</td>
<td>21</td>
<td>25</td>
<td>37</td>
<td>93</td>
<td>22</td>
<td></td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Session 5</td>
<td></td>
<td></td>
<td>52</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Risk Score:
- Session 1: 0.226
- Session 2: 0.114
- Session 3: 0.092
- Session 4: 0.007
- Session 5: 0.981
Location Risk
• - lat-lon coordinates from device != lat-lon from IP-address
• - lat-lon changing faster than plane speed ~800km/hour

Money Movement Risk
• - Unusual amounts of money transfer
• - Unusual directions of money transfer

Device Risk
• - New device never used before
• - Unusual device combination, based on total device matrix

Time Risk
• - Unusual time of day for current customer
• - Unusual time of day for all customers
Strengths of the fraud detection tool

- ML Model for analyzing user behavior to detect out of pattern activity.
- Contains a system for learning from the feedback - if you return to model results marked as fraud or falsely identified fraud.
- It analyses based on the individual user’s behavior.
- Begins to work even on a small amount of data (the problem of a cold start is solved), does not require a large number of cases of confirmed fraud.
- Done in real time on a session to stop fraud before it happens not after the fact historical review.
- We can monitor multiple channels, mobile, online and others.
- Configurable thresholds for Accept, Review, Stop transaction.
PART II: CNTK + C# IN REAL LIFE PROJECTS: FRAUD DETECTION

Detection

From: 09/01/2018 12:00 AM  To: 09/27/2018 12:00 AM

Session Risk Level:

- Show Advanced Search

Save Session Details

Save

Clear all

Search results: 70 sessions (0 selected):

- User: Anakin
  - Session Start: 09/26/2018 07:37:31 AM
  - Session Duration: 00:00
  - Session Risk Score: 70

- User: Anakin
  - Session Start: 09/26/2018 07:20:10 AM
  - Session Duration: 05:00
  - Session Risk Score: 100

- User: Anakin
  - Session Start: 09/26/2018 07:11:09 AM
  - Session Duration: 01:56
  - Session Risk Score: 100

- User: Anakin
  - Session Start: 09/26/2018 07:08:59 AM
  - Session Duration: 00:51
  - Session Risk Score: 60

- User: Anakin
  - Session Start: 09/26/2018 07:06:06 AM
  - Session Duration: 01:10
  - Session Risk Score: 100

- User: Skywalker
  - Session Start: 09/26/2018 04:39:11 AM
  - Session Duration: 05:50
  - Session Risk Score: 100
PART II: CNTK + C# IN REAL LIFE PROJECTS: FRAUD DETECTION

Risk Score Threshold:
- **High risk**: 95 sessions by 3 months (0.5% of all sessions), ~3 sessions per day
- **Medium risk**: 65 sessions by 3 months (12% of all sessions), ~71 sessions per day
- **Low risk**: 40 sessions by 3 months (18% of all sessions), ~107 sessions per day
- **No risk**: 4,962 sessions by 3 months (69.5% of all sessions), ~414 sessions per day
Part I: how we choose CNTK + c#

- Access Softek company, our tasks, data volumes
- 3 pillars of our requirements: speed, fast and cheap training and accuracy
- Comparison CNTK with other existing solutions

Part II: CNTK + c# in real life projects

- Fraud detection
- User Engagement
- Churn Prevention
Engagement rate calculation

1) We have 37 features like session_interval, n_login_fails_login etc. Based on the received features we calculate the values of 3 dimensions.

\[
\text{Frequency/recency} = \frac{1}{1+e^{-\left(\sum f_i \cdot \log_2 f_i\right)}} \quad (1) \quad \text{Depth or intensity} = \frac{1}{1+e^{-\left(\sum f_i \cdot \log_2 f_i\right)}} \quad (2) \quad \text{Money movement} = \frac{1}{1+e^{-\left(\sum f_i \cdot \log_2 f_i\right)}} \quad (3)
\]

2) When the values of Frequency-Recency, Depth-Of-Intensity, Money are calculated, we form the preTarget variable as Frequency-Recency + Depth-Of-Intensity + Money and sort the entire array of data by this value. We take for training the model only 20% of the sample.

3) 10% of the top in our preTarget variable is denoted as the Target variable = 1, 10% of the bottom by the preTarget variable is denoted as the Target variable is 0. This data is submitted to the input for model training.

4) For the forecast whole set of calculated features is submitted to the input of the model, and the Engagement Rate for the user is calculated for each user.
PART II: CNTK + C# IN REAL LIFE PROJECTS: USER ENGAGEMENT

Average Engagement Rate
Average and standard deviation of the Engagement Rate among all users

Engagement Rate description

- Average engagement rate
- Std deviation
## User segmentation by Engagement

This chart displays segmentation of users based on their level of engagement with your app over the chosen time period.

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIP</td>
<td>Users who use the application frequently (and recently) and use many features</td>
</tr>
<tr>
<td>Growing</td>
<td>Users who use the app periodically but only for some features</td>
</tr>
<tr>
<td>Newcomers</td>
<td>Users who use the app frequently but only for a few features</td>
</tr>
<tr>
<td>Lost</td>
<td>Users who previously used the many app features but rarely use the app lately</td>
</tr>
<tr>
<td>One-off</td>
<td>Users who previously used the app and didn’t use many features</td>
</tr>
</tbody>
</table>

### Engagement segments calculation

Users are classified into segments to a specific date using the following logic:

**Frequency of use:** This property indicates how frequently the user uses the app. It is calculated based on the following factors:

- average interval between sessions (for the previous [day, week, month])
- average session duration (for the previous [day, week, month])
- time since the user’s last login (recency)

The possible values for each factor are 1, 2, or 3 where “3” represents high frequency and “1” represents low frequency.

**Depth of usage:** This property shows how deeply each user uses the app based on the number of features utilized in a session. It is calculated based on the following factors:

- number of unique events (for the previous [day, week, month])
- average time spent on one event (per [day, week, month] before the date)
- average, min and max event frequency - how often the event happens compared to other events (for the previous [day, week, month])
- variance of event frequency - deviation from the usual pattern of event frequency for the user (for the previous [day, week, month])
- number of devices used by the user (for the previous month)
- number of users per device (for the previous month)
- number of time the app was deleted and reinstalled (for the previous month)

The possible values for each factor are 1, 2, or 3 where “3” represents deep usage of the app and “1” represents superficial usage of the app.
### User segmentation by Engagement

This chart displays segmentation of users based on their level of engagement with your app over the chosen time period.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIP</td>
<td>Users who use the application frequently (and recently) and use many features</td>
</tr>
<tr>
<td>Growing</td>
<td>Users who use the app periodically but only for some features</td>
</tr>
<tr>
<td>Newcomers</td>
<td>Users who use the app frequently but only for a few features</td>
</tr>
<tr>
<td>Lost</td>
<td>Users who previously used the many app features but rarely use the app lately</td>
</tr>
<tr>
<td>One-off</td>
<td>Users who previously used the app and didn’t use many features</td>
</tr>
</tbody>
</table>
Migration 75 user(s) from segment "VIP" to segment "Growing"

Time since the user’s last login (recency) decreased on 121.18%

Min event frequency for the previous day decreased on 6.71%

Min event frequency for the previous week decreased on 6.71%

Min event frequency for the previous month decreased on 6.71%

Average event frequency for the previous day decreased on 6.05%
Migration 31 user(s) from segment “One-off” to segment “VIP”

Number of unique events for the previous day increased on 12.83%

Number of unique events for the previous week increased on 12.83%

Number of unique events for the previous month increased on 12.83%
Part I: how we choose CNTK + c#

- Access Softek company, our tasks, data volumes
- 3 pillars of our requirements: speed, fast and cheap training and accuracy
- Comparison CNTK with other existing solutions

Part II: CNTK + c# in real life projects

- Fraud detection
- User Engagement
- Churn Prevention
PART II: CNTK + C# IN REAL LIFE PROJECTS: CHURN PREVENTION
PART II: CNTK + C# IN REAL LIFE PROJECTS: CHURN PREVENTION
PART II: CNTK + C# IN REAL LIFE PROJECTS: CHURN PREVENTION
### Count of Member Number by Date and Probability Of Churn

<table>
<thead>
<tr>
<th>Probability of Churn</th>
<th>Sep 14, 2018</th>
<th>Sep 13, 2018</th>
<th>Sep 12, 2018</th>
<th>Sep 11, 2018</th>
<th>Sep 10, 2018</th>
<th>Sep 9, 2018</th>
<th>Sep 8, 2018</th>
<th>Sep 7, 2018</th>
<th>Sep 6, 2018</th>
</tr>
</thead>
<tbody>
<tr>
<td>96</td>
<td>55</td>
<td>55</td>
<td>52</td>
<td>49</td>
<td>52</td>
<td>30</td>
<td>45</td>
<td>86</td>
<td>59</td>
</tr>
<tr>
<td>95</td>
<td>61</td>
<td>48</td>
<td>50</td>
<td>40</td>
<td>54</td>
<td>40</td>
<td>36</td>
<td>60</td>
<td>52</td>
</tr>
<tr>
<td>94</td>
<td>64</td>
<td>57</td>
<td>61</td>
<td>66</td>
<td>37</td>
<td>30</td>
<td>43</td>
<td>54</td>
<td>59</td>
</tr>
<tr>
<td>93</td>
<td>56</td>
<td>61</td>
<td>46</td>
<td>42</td>
<td>54</td>
<td>53</td>
<td>46</td>
<td>80</td>
<td>55</td>
</tr>
<tr>
<td>92</td>
<td>51</td>
<td>65</td>
<td>51</td>
<td>62</td>
<td>61</td>
<td>53</td>
<td>26</td>
<td>67</td>
<td>66</td>
</tr>
<tr>
<td>91</td>
<td>57</td>
<td>52</td>
<td>61</td>
<td>51</td>
<td>52</td>
<td>46</td>
<td>54</td>
<td>71</td>
<td>46</td>
</tr>
<tr>
<td>90</td>
<td>42</td>
<td>46</td>
<td>78</td>
<td>68</td>
<td>56</td>
<td>37</td>
<td>47</td>
<td>61</td>
<td>60</td>
</tr>
<tr>
<td>89</td>
<td>52</td>
<td>59</td>
<td>48</td>
<td>56</td>
<td>24</td>
<td>50</td>
<td>58</td>
<td>61</td>
<td>61</td>
</tr>
<tr>
<td>88</td>
<td>56</td>
<td>64</td>
<td>62</td>
<td>59</td>
<td>53</td>
<td>40</td>
<td>48</td>
<td>98</td>
<td>52</td>
</tr>
<tr>
<td>87</td>
<td>50</td>
<td>56</td>
<td>48</td>
<td>58</td>
<td>51</td>
<td>46</td>
<td>41</td>
<td>60</td>
<td>61</td>
</tr>
<tr>
<td>86</td>
<td>44</td>
<td>43</td>
<td>43</td>
<td>52</td>
<td>52</td>
<td>42</td>
<td>41</td>
<td>54</td>
<td>60</td>
</tr>
<tr>
<td>85</td>
<td>48</td>
<td>46</td>
<td>56</td>
<td>52</td>
<td>56</td>
<td>33</td>
<td>41</td>
<td>80</td>
<td>58</td>
</tr>
<tr>
<td>84</td>
<td>80</td>
<td>60</td>
<td>52</td>
<td>50</td>
<td>57</td>
<td>31</td>
<td>61</td>
<td>69</td>
<td>44</td>
</tr>
<tr>
<td>83</td>
<td>48</td>
<td>52</td>
<td>53</td>
<td>62</td>
<td>48</td>
<td>41</td>
<td>45</td>
<td>75</td>
<td>65</td>
</tr>
<tr>
<td>82</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
QUESTIONS? COMMENTS? CONCERNS?