Pixie: A system for Recommending 3+ Billion Items to 200+ Million Users in Real-Time

Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, Jure Leskovec
What is Pinterest

- Visual catalog
- Pins: visual bookmarks
- Boards: curated by users collections of similar pins
- 3+ billion items
- 200+ million monthly active users
Outline

● Problem Statement
● Innovation/contributions of the proposed work
● Related Work
● Proposed Method
● Evaluation Methods
● Conclusion
The Pinterest Problem

The provided recommendations must be:

- Personalized
- Engaging
- Scalable
  - Billions of items
  - In real-time
Proposed method: Pixie

- Scalable, graph-based, real-time RS
- Given a set of user-specific pins as a query, Pixie selects in real-time from billions of pins those that are most related to the query
- Generates recommendations by using the Pinterest object graph
  - 3 billion nodes
  - 17 billion edges
Pixie Innovations

- Bias the algorithm in a user-specific way
- Multiple query pins with different importance
- Boost recommendations related to multiple query pins
- Early stopping
- Cold-start problem via recommendations of both pins and boards
- Graph curation

Advantages

- Varying walk length
- Handle smaller boards
- Constant time
Related Work

- Web-scale recommender systems
  - Not real-time, precomputed recommendations
  - Real-time news recommendations
- Random-walk-based approaches
  - “Who to follow” at Twitter
- Traditional collaborative filtering approaches
  - Time and space complexity restrictions
- Content-based methods
  - Deep learning
The Pinterest Object Graph

- Pins, Boards
- Bipartite graph $G = (P, B, E)$
- There is an edge $e \in E$ between a pin $p \in P$ and a board $b \in B$ if a user saved $p$ to $b$
- $E(p)$
- $E(b)$
- Query set $Q = \{(q, w_q)\}$
Algorithm 1 Basic Random Walk; $q$ is the query pin; $E$ denotes the edges of graph $G$; $\alpha$ determines the length of walks; $N$ is the total number of steps of the walk; $V$ stores pin visit counts.

\begin{algorithm}
\hspace*{1em}\textsc{BasicRandomWalk}(q: \text{Query pin}, E: \text{Set of edges}, \alpha: \text{Real}, N: \text{Int})$
\begin{algorithmic}
\STATE totSteps = 0, $V = \emptyset$
\REPEAT
\STATE currPin = $q$
\STATE currSteps = SampleWalkLength($\alpha$)
\FOR {i = [1 : currSteps]}
\STATE currBoard = $E$(currPin)[rand()]
\STATE currPin = $E$(currBoard)[randNeighbor()]
\STATE $V$[currPin]++
\ENDFOR
\STATE totSteps += currSteps
\UNTIL totSteps $\geq$ $N$
\RETURN $V$
\end{algorithmic}
\end{algorithm}
Biasing the Pixie Random Walk

- Personalization
- User features
  - Different languages
  - Different topics
- Biased edge selection
Multiple Query Pins with Weights

- **Weights based on**
  - The time since the last interaction
  - The type of interaction

- **Step distribution:**
  - Scaling of the number of steps allocated to each query pin to be proportional to its degree
  - Pins with low degrees receive sufficient number of steps

\[
s_q = |E(q)| \cdot (C - \log |E(q)|)
\]

\[
N_q = w_q N \frac{s_q}{\sum_{r \in Q} s_r}
\]
Multi-hit Booster

- Prefer recommendations that are related to multiple query pins in $Q$
  - More relevant to the entire query
- Pixie boosts the scores of candidate pins that are visited from multiple query pins
- $V[p]$ is the combined visit count for pin $p$ over all the query pins $q \in Q$

$$V[p] = \left( \sum_{q \in Q} \sqrt{V_q[p]} \right)^2$$
Early Stopping

- The runtime depends on the number of steps
- Instead of having a fixed number of steps for all query pins, the number of steps depends on the query
- $n_p$ and $n_v$:
  terminate the walks when at least $n_p$ candidate pins have been visited at least $n_v$ times
Pixie Random Walk Algorithm

**Algorithm 2** Pixie Random Walk algorithm with early stopping.

\[
\text{PIXIERANDOMWALK}(q: \text{Query pin}, E: \text{Set of edges}, U: \text{User personalization features}, \alpha: \text{Real}, N: \text{Int}, n_p: \text{Int}, n_v: \text{Int})
\]

1: totSteps = 0, V = φ
2: nHighVisited = 0
3: repeat
4:    currPin = q
5:    currSteps = SampleWalkLength(\alpha)
6:    for i = [1 : currSteps] do
7:        currBoard = E(currPin)[PersonalizedNeighbor(E, U)]
8:        currPin = E(currBoard)[PersonalizedNeighbor(E, U)]
9:        V[currPin]++
10:       if V[currPin] == n_v then
11:           nHighVisited++
12:       totSteps += currSteps
13:    until totSteps ≥ N or nHighVisited > n_p
14: return V

**Algorithm 3** Pixie recommendations for multiple pins.

\[
\text{PIXIERANDOMWALKMULTIPLE}(Q: \text{Query pins}, W: \text{Set of weights for query pins}, E: \text{Set of edges}, U: \text{User personalization features}, \alpha: \text{Real}, N: \text{Int})
\]

1: for all \( q \in Q \) do
2:    \( N_q = \text{Eq. 2} \)
3:    \( V_q = \text{PIXIERANDOMWALK}(q, E, U, \alpha, N_q) \)
4: for all \( p \in G \) do
5:    \( V[p] = \left( \sum_{q \in Q} \sqrt{V_q[p]} \right)^2 \)
6: return \( V \)
Graph Pruning

- Improved recommendation quality
  - Topically focused graph
  - Smaller graph
- Diverse boards are removed
- Edges that correspond to miscategorized pins are discarded
- Pruning factor $\delta$
- Pruned graph:
  - 1 billion boards
  - 2 billion pins
  - 17 billion edges
Experimental Evaluation
Pixie Recommendation Quality

- Ranking the Most Related Pin: *Given a user, we want to predict which pin she will engage with next.*
- User activity: save
- Hit rate

<table>
<thead>
<tr>
<th>Method</th>
<th>Top 10</th>
<th>Top 100</th>
<th>Top 1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-based (textual)</td>
<td>1.0%</td>
<td>2.2%</td>
<td>4.8%</td>
</tr>
<tr>
<td>Content-based (visual)</td>
<td>1.1%</td>
<td>2.4%</td>
<td>4.5%</td>
</tr>
<tr>
<td>Content-based (combined)</td>
<td>2.1%</td>
<td>4.6%</td>
<td>10.5%</td>
</tr>
<tr>
<td>Pixie (graph-based)</td>
<td>6.3%</td>
<td>23.1%</td>
<td>52.2%</td>
</tr>
</tbody>
</table>

Table 1: Given a query pin, predict which pin will be repinned. Performance is quantified by fraction of times the correct pin was ranked among top $K$. 
Pixie Recommendation Quality

- A/B Experiments
- User engagement: what is the fraction of pins that a user engages in by clicking, liking or saving them

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Lift</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homefeed, per pin engagement</td>
<td>+48%</td>
</tr>
<tr>
<td>Related pins, per pin engagement</td>
<td>+13%</td>
</tr>
<tr>
<td>Board recommendations, per pin engagement</td>
<td>+26%</td>
</tr>
<tr>
<td>Localization, pins in user local language</td>
<td>+48-75%</td>
</tr>
<tr>
<td>Explore tab, per pin engagement</td>
<td>+20%</td>
</tr>
</tbody>
</table>

Table 2: Summary of A/B experiments across different Pinterest user surfaces. Lift in engagement of Pixie vs. current production systems.
Figure 1: (a) Runtime of $\text{PIXIE}_\text{RANDOMWALK}$ against the number of steps and (b) against the size of the query set.
Variance of Top Results

Figure 2: The variance of results against number of steps.
# Evaluation of the Biased Walk

<table>
<thead>
<tr>
<th></th>
<th>En→Japanese</th>
<th>Japanese→Japanese</th>
<th>En→Spanish</th>
<th>Spanish→Spanish</th>
<th>En→Slovak</th>
<th>Slovak→Slovak</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BasicRandomWalk</strong></td>
<td>16.35%</td>
<td>52.95%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>PixieRandomWalk</strong></td>
<td>80.33%</td>
<td>100.00%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>BasicRandomWalk</strong></td>
<td>41.94%</td>
<td>74.02%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>PixieRandomWalk</strong></td>
<td>94.51%</td>
<td>100.00%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>BasicRandomWalk</strong></td>
<td>2.13%</td>
<td>16.06%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>PixieRandomWalk</strong></td>
<td>42.55%</td>
<td>100.00%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Comparison of the proportion of target-language content produced by **BasicRandomWalk** and **PixieRandomWalk**. The second column shows the percentage of candidates in the target language when the query pin is in the English language and the third column shows the percentage when the query pin itself is in the target language.
Early stopping

Figure 3: (a) Early stopping performance against $n_v$ with $n_p = 2,000$. (b) Early stopping performance against $n_p$ with $n_v = 4$. 
Evaluation of Graph Pruning

Figure 4: F1 scores for link prediction and number of edges for different graph pruning factors.

Figure 5: The memory usage and Pixie runtime against different pruned graphs.
Take away messages

● Responsive real-life applications
● Time and space complexity of the proposed methods matter
● Random walks work
  ○ High performance
  ○ Scalability
Thank you!