Cross-media measurement with Virtual People

Evgeny Skvortsov, Jim Koehler

Google LLC

May 2021

Abstract

We introduce methods for efficient and privacy-safe modeling of reach and the demographic composition of cross-media campaigns. Cross-media campaign traffic is composed of two parts: digital and TV. Digital traffic is estimated based on event-level data available in server logs. TV traffic is extrapolated from a combination of panel and set-top-box or smart-TV data. The Virtual-People methodology introduced in [9] allows for efficient measurement of digital audiences. In this paper we extend this methodology to work with the extrapolated data sources associated with TV data, thus generalizing it for cross-media measurement.

1 Introduction

Koehler, Skvortsov, and Vos (2013) [4] (KSV) presents a method for measuring the reach and frequency of online-ad campaigns by audience attributes for one device (or cookie) type. This method combines ad server logs, publisher provided user data (PPD), census data, and a representative panel to produce corrected cookie and impression counts by these audience attributes. The method corrects for cookie issues such as deletion and sharing, and for PPD issues such as non-representativeness and poor quality of demographic labels. It also proposes a model that converts cookie counts to user counts.

Koehler, Skvortsov, Ma, Liu (2016) [3] (KSML) extends the method to today's world of multiple device types such as desktop, smartphone, and tablet. A formulation for converting multiple cookie counts to people counts is proposed. The article introduced the concept of an Activity Distribution Function (ADF), which describes the probability of a person generating cookies of each type. A theory relating ADFs to matching cross-device reach functions is presented and shows that ADFs can be well approximated by a mixture of Dirac delta functions [1] which can be estimated empirically using panel data. A natural extension of the demographic correction to multiple devices is presented as well.

Skvortsov and Koehler (2019) [9] (KS) presents a technology that implements the methodologies of [4] and [3] in a large scale production systems efficiently. The reach and demographic-correction models are converted into assignments of virtual people to each of the events in the logs. Each virtual person has demographic attributes (age and gender) assigned to them. The total reach of an audience (ad campaign, web site, online video etc) can be estimated as a simple count of unique virtual people assigned to the corresponding set of events. The demographic composition of an audience is estimated as the demographic composition of the set of virtual people.

Kreuter et al (2020) [5] describes a multi-party cryptographic protocol that adds differentially private noise for calculating cross-publisher Virtual-People reach. The protocols are analyzed theoretically describing the level of user-privacy protection. These protocols work by data providers exchanging so called Liquid Legions sketches. These are cardinality sketches, encoding sets of Virtual People identifiers and are designed to be compatible with further processing in a multi-party computation for the calculation of total reach in a way where workers have only a limited exposure of the information about individual users.

Tsai, Skvortsov and Koehler (2021) [2] describe algorithms for efficiently creating HyperLogLog sketches of Virtual People corresponding to panelists. The algorithm works by creating a so called *Deep HyperLogLog* sketch of the audience. The size of the deep sketch is in practice an order magnitude smaller than the size of the original population and this sketch allows creating HyperLogLog sketches of random samples of the audience in time linear to the size of the deep sketch.

This paper extends the technologies of [9], [2] and [5] to allow efficient reporting of the reach of cross-media, cross-publisher audiences in a way that preserves the privacy of individual users. The main algorithmic contribution of this paper is an efficient method for encoding audience estimates obtained by an extrapolation into LiquidLegions sketches of Virtual People. With this method, reach estimates of TV audiences measured via an extrapolation from a panel or from partial set-top-box data can be encoded in a sketch and serve as an input to the system described in [5]. The system naturally deduplicates TV reach from the digital part of the audience via a simple sketch merge.

The rest of the paper is organized as follows. In Section 2 we describe the methodological approaches to measuring TV audiences with Virtual People: aggregate level that is compatible with the arbitrary legacy currencies and the per-event approach, which has advantage of full self consistency, but is more susceptible to noise if panel weights have high volatility. In Section 3 we describe algorithms for encoding the reach measurement obtained from panel into Liquid Legions sketches, which are compatible with the system described in [5].

2 Mapping Audiences to Sets of Virtual People

Server logs are the primary source of information when a digital audience is being measured. Server logs record all of the individual events through which an audience is exposed to an ad campaign. The Methodology described in [9] assigns Virtual Person identifiers to individual events. Reach for a set of events can then be estimated by counting the number of unique Virtual People identifiers assigned to these events.

On the other hand, TV audience measurement usually lacks comprehensive information about individual view events. It is usually performed via extrapolation from panel data, or from data collected from a subset of set-top-boxes and/or smart TVs. In this section we describe two approaches to how a measurement done from this sample can be encoded into sets of virtual people. These two approaches are

1. **Aggregate approach**: A traditional panel-based measurement of a campaign consists of aggregated reach estimates for each demographic bucket. The **Aggregate** approach then samples an appropriate amount of virtual people from the total virtual population to approximately match the aggregate estimates.

2. **Per event extrapolation approach**: Each event is hashed to a set of Virtual People. Audience reports are then the union of all virtual people that were assigned to the collection of events associated with the campaign/report. Reach is the sum of these virtual people.

In both approaches the Virtual People can be stored in a sketch and transmitted into the cross media measurement system's secure cardinality estimation framework to be unionized with the digital reach.

The advantage of the Aggregate approach is its compatibility with an arbitrary TV audience measurement methodology in that it simply encodes it for deduplication with its associated digital reach. It can thus be used for blending existing *currency* TV numbers with the state-of-the-art digital reach estimates. This approach is appropriate for a system that produces reach estimates at the end of a campaign as it cannot guarantee consistency at different times during the campaign.

On the other hand, the per-event approach guarantees consistency for TV measurement and can also be implemented with a simpler engineering infrastructure. While it may not provide exact agreement with existing TV numbers, we encourage considering it when developing new TV reach measurement systems.

From a modeling point of view, in both approaches the overlap between digital and TV audiences is determined by collisions of Virtual People identifiers sampled from pre-defined population pools. The population can be broken into these pools by using census categories, such as demographic age/gender buckets, or geography regions. In addition to categories important for reporting this partition could be using activity-based characteristics, such as light/heavy TV viewership. The sub-division of census based pools into pools of activity-based characteristics can be done with the goal of optimising resulting overlap model accuracy.

Next we formally define these two approaches.

2.1 Aggregate Virtual People Mapping

Let C be the set of people categories for which reach reports are being computed. For instance C can be the set of demographics, i.e. each $c \in C$ could be a pair of gender and a 10 year age bucket. Geography, or interests could also be part of the people category definition.

A reach report is then defined as a function that maps each people category to the number of people reached in this category. Formally as a map $R : C \mapsto \mathbb{Z}$. The Aggregate mapping approach is to map this report to the set of virtual people. Algorithm 1 presented below shows how to build this map in time linear with respect to the total number of virtual people.

input : Set of Virtual People broken by category $\mathcal{V} = \bigcup_{c \in C} \mathcal{V}_c$, reach report $R : C \mapsto \mathbb{Z}$
output : A set of virtual people identifiers $A \subseteq \mathcal{V}$ representing report R
let $A = \varnothing$;
for $c \in C$ **do**
  for $p \in \mathcal{V}_c$ **do**
    with probability $\frac{R(c)}{|V_c|}$ add p to A
  end
end

Algorithm 1: Mapping aggregate reach report to a set of Virtual People

Observation 1 *If a set A is built by Algorithm 1 for report R, then we have $|A \cap \mathcal{V}_c| \approx R(c)$.*

Observation 2 *Runtime complexity of the Algorithm 1 is $O(|\mathcal{V}|)$.*

2.2 Per-event Virtual People assignment

Aggregate methods of measuring reach from a panel are not necessarily consistent. As an example, consider the situation where a subset of panelists are exposed to a campaign during its initial phase and no other panelists are exposed subsequently. If the weights for these exposed panelists decrease over the campaign time-interval, then the campaign reach estimates will also decrease over that time.

Assigning virtual people to individual events of the panel or to set-top-box data ensures that reach estimates are internally consistent. This is highly desirable for reach planning scenarios and also allows continuous tracking and adjustment of campaign performance.

It is not always possible to implement an aggregate approach with per-event assignment. For example, most aggregate approaches use panelist's weights for an given period - usually the interval of the campaign. As the set of panelists changes over time their respective weights will also change. This means that the contribution of a given event to reach depends on the time interval being considered and hence such a method is not equivalent to any per-event assignment.

There many measurement techniques that can be implemented with the assignment of virtual people to panel events. Below we describe one of these approaches, which we find to be reasonable.

input : An item i, a list of options ℓ and an assignment of weights of the options ω.
output : A consistent hash $h \in \ell$ of the item i
let $r = null$;
let $h = \infty$;
for $j \in \ell$ **do**
  let x be a float valued fingerprint of pair (i, j) from interval $(0, 1)$;
  let $h' = -\omega(j) \cdot \log x$;
  if $h' < h$ **then**
    update $r = j$;
    update $h = h'$;
end
return h;

Algorithm 2: *AffinityHash*(i, ℓ, ω)

To handle any changes in panel weights, we use the affinity hashing described in [6]. Algorithm 2 is the fundamental building block for the mapping of panel events to Virtual People. Algorithm 3 uses affinity hashing to assign each Virtual Person to a panelist within the same category represented by that Virtual Person. Thus each panelist is allocated a set of Virtual People.

Observation 3 *The runtime complexity of Algorithm 3 is* $O(\sum_c |\mathcal{V}_c| \cdot |P_c|)$.

Algorithm 4 applies Algorithm 3 for each day, thus minimizing the change in Virtual People assigned to any panelists, as the set of panelists, or the weights of panelists change from day to day.

3 Efficient LiquidLegions sampling

In this section we discuss more efficient alternatives to Algorithms 1 and 3. The runtime complexity of these algorithms are $O(|\mathcal{V}|)$ and $O(\sum_c |\mathcal{V}_c| \cdot |P_c|)$, respectively, and have a multiplier of the total number of virtual people $|\mathcal{V}|$. Rather that running over the whole set of Virtual People we can

input : Set of Virtual People labeled by category $\mathcal{V} = \bigcup_{c \in C} \mathcal{V}_c$, a set of panelists labeled by the same categories $P = \bigcup_{c \in C} P_c$, a set of panelist weights $\omega : P \mapsto \mathbb{R}$

output : A mapping from panelists to sets of virtual people $\tilde{V} : P \mapsto 2^{\mathcal{V}}$.

let $\tilde{V}(p) = \varnothing$ for all $p \in P$;

for $c \in C$ **do**

  for $v \in V$ **do**

    let $p = \mathit{AffinityHash}(v, P_c)$;

    update $\tilde{V}(p) = \tilde{V}(p) \cup v$;

  end

end

Algorithm 3: Association of Panelists with Virtual People.

input : Set of Virtual People broken by category $\mathcal{V} = \bigcup_{c \in C} \mathcal{V}_c$, a set of panelists broken by the same categories $P = \bigcup_{c \in C} P_c$, a set of events broken by date $\mathcal{E} = \bigcup_{d \in D} \mathcal{E}_d$, correspondence of events to panelists $p : \mathcal{E} \mapsto P$ and panel weights for each day $\omega : P \times D \mapsto \mathbb{R}$

output : A mapping from events to sets of virtual people $V : \mathcal{E} \mapsto 2^{\mathcal{V}}$.

let $\tilde{V}_d$ be obtained by Algorithm 3 from weights of panelists on day d;

for $d \in D$ **do**

  for $e \in \mathcal{E}$ **do**

    let $V(e) = \tilde{V}_d(p(e))$

  end

end

Algorithm 4: Mapping panel events to sets of Virtual People

use an auxiliary structure that we call Deep Liquid Legions sketch. This bring downs the runtime complexity of Algorithms 1 and 3 to $O(|\mathcal{S}|)$ and $O(\sum_c |\mathcal{S}_c| \times |P_c|)$ respectively, where $|\mathcal{S}|$ is the size of the Deep Liquid Legions sketch. The size of the sketch depends on the desired relative error of the approximate cardinality estimation. By using simulation we observe that a sketch size of around $100K$ adds less than 2% relative error. For the United States, demographic buckets usually contains tens of millions of Virtual People, which means that using Deep Liquid Legion sketches of size $100K$ speed up the algorithms by roughly hundred times.

As it was described in [5], LiquidLegions sketch with parameters (m, α) is a Bloom Filter over m registers, which uses a truncated exponential distribution with decay α for allocating items to these registers.

Definition 1 Depth *of the LiquidLegions sketch, δ, is the index of the first position of the sketch that is occupied by 0.*

Definition 2 Deep Liquid Legions *sketch of the set S, with parameters m, α, is a triple $\mathcal{S} = (L, \phi, \psi)$, where L is the LiquidLegions sketch with parameters m and α, of the set S with two additional functions:*

- *Map $\phi : [\delta + 1, \ldots, m - 1] \mapsto 2^S$, where $\phi(i)$ is the set of objects from S that got assigned to position i.*
- *Map $\psi : [0, \ldots, \delta - 1] \mapsto S$, where $\psi(i)$ is one of the objects that got assigned to position i.*

Where δ is the depth of the sketch L.

We use notation $|\mathcal{S}|$ for the total number of elements in the values of functions ϕ and ψ, i.e. $|\mathcal{S}| = \delta + \sum_i |\phi(i)|$. We call this value *size* of the sketch $\mathcal{S}$. Indeed to store Deep LiquidLegions sketch we need to store the maps ϕ and ψ. Therefore amount of memory required for storing the sketch is proportional to $|\mathcal{S}|$.

Algorithm 5 shows how to construct the deep sketch of a set of Virtual People. This algorithm runs in time that is linear with respect to the total number of Virtual People.

Deep sketch of a set $\mathcal{V}$ can be used for efficient creation of sketches for subsets of $\mathcal{V}$. In the next subsections we describe how the deep sketch can be used for the aggregate and the per-event modeling approaches. For simplicity, we will describe them under the assumption of a single set $\mathcal{V}$ to subdivide. This corresponds to deduping TV and digital measurement via a full independence assumption. To dedupe via independence conditional on categories we need to apply these algorithms separately to $\mathcal{V}_c$ for each category c, e.g. for every age/gender demographic bucket.

3.1 Sketch for the aggregate approach

For the aggregate approach, a TV reach measurement that is computed via a panel needs to be encoded into a set of Virtual People and then passed to a secure cardinality aggregation protocol to be deduplicated with its companion digital measurement of the campaign.

Algorithm 6 describes how the set of Virtual People of cardinality n can be built from the deep sketch of the audience. Note that the deep sketch of the population can be built once and can be re-used for all queries over the given population. Algorithm 6 is to be applied each time a cross media reach report needs to be generated.

Observation 4 *The runtime complexity of Algorithm 6 is $O(|\mathcal{S}|)$.*

input : A set of Virtual People $\mathcal{V}$, truncated exponential allocation of Virtual People to LiquidLegions registers $r(v)$
output : Deep LiquidLegions sketch for the set $\mathcal{V}$
let $S =$ bit array of length m pre-populated with 0;
let $\phi(i) = \varnothing$ for all $i \in [0, \ldots, m-1]$;
for $v \in \mathcal{V}$ **do**
  let $i = r(v)$;
  update $S[i] = 1$;
  update $\phi(i) = \phi(i) \cup v$;
end
let $\delta = \min_{\phi(i)=\varnothing} i$;
for $i \in [0, \ldots, \delta - 1]$ **do**
  let v be a randomly sampled element from $\phi(i)$;
  let $\psi(i) = v$
end
restrict domain of ϕ to $[\delta + 1, \ldots, m-1]$;
return (S, ϕ, ψ)

Algorithm 5: Building Deep LiquidLegions sketch for a set of Virtual People

input : A Deep LiquidLegins sketch $\mathcal{S} = (S_0, \phi, \psi)$ with parameters (m, α) of a set $\mathcal{V}$ of depth δ, and cardinality $n < |\mathcal{V}|$
output : A sketch S representing a random subset of $\mathcal{V}$ of cardinality close to n
let $f = \frac{n}{|\mathcal{V}|}$;
let S be empty sketch of length m;
for $i \in [\delta + 1, \ldots, m-1]$ **do**
  for $v \in \psi(i)$ **do**
    with probability f set $S[i] = 1$;
  end
end
let $h = |\mathcal{V}| - |\{x | \exists i \; x \in \phi(i)\}|$;
for $i \in [0, \ldots, \delta - 1]$ **do**
  let $x = i/m$;
  with probability $\left(1 - \frac{a \cdot e^{-ax}}{(1-e^{-a}) \cdot m}\right)^{h \cdot f}$ set $S[i] = 1$;
end
return S

Algorithm 6: Using Deep LiquidLegions sketch to create a sketch for a random subset of given cardinality.

3.2 Sketch for the per-event approach

To apply efficient sketching for the per-event approach we need to implement splitting the audience deep sketch into sketches of disjoint sets, which corresponds to the Algorithm 3. Algorithm 7 performs this split operation on the sketches.

Observation 5 *The runtime complexity of Algorithm 7 is* $O(|\mathcal{S}| \times |P|)$.

input : A Deep LiquidLegins sketch (S_0, ϕ, ψ) with parameters (m, α) of a set $\mathcal{V}$ of depth δ, cardinalities $n_1, \ldots, n_K, \sum_{k=1}^{K} n_k = |\mathcal{V}|$.
output : Sketches $S_1, \ldots, S_k$, representing partition of $\mathcal{V}$ into subsets of cardinalities $n_1, \ldots, n_K$.

let $f_k = \frac{n_k}{|\mathcal{V}|}$ for $k \in 1, \ldots, K$;
let S_k be empty sketch of length m for $k \in 1, \ldots, K$;
for $i \in [\delta + 1, \ldots, m - 1]$ **do**
  for $v \in \phi(i)$ **do**
    let $k = \mathit{AffinityHash}(v, \{1, \ldots, k\}, \{f_1, \ldots, f_k\})$;
    update $S_k[i] = 1$;
  end
end
for $i \in [0, \ldots, \delta - 1]$ **do**
  let $k = \mathit{AffinityHash}(\psi(i), \{1, \ldots, k\}, \{f_1, \ldots, f_k\})$;
  update $S_k[i] = 1$;
end
let $h = |\mathcal{V}| - |\{x | \exists i \; x \in \phi(i)\}| - \delta$;
for $i \in [0, \ldots, \delta - 1]$ **do**
  let $x = i/m$;
  for $k \in [1, \ldots, K]$ **do**
    let $\rho = \left(1 - \frac{a \cdot e^{-ax}}{(1 - e^{-a}) \cdot m}\right)^{h \cdot f_k}$;
    let $z = \mathit{AffinityHash}(k, \{1, 0\}, \{\rho, 1 - \rho\})$;
    if $z = 1$ **then**
      update $S_k[i] = 1$;
  end
end
return $\{S_1, \ldots, S_K\}$

Algorithm 7: Using Deep LiquidLegions sketch to create sketches for a random disjoint partition of given cardinalities.

4 Conclusion

We presented approaches to modeling cross-media reach within the Virtual People framework. The digital part of the audience is measured with the standard approach proposed in [9]. For the TV part of the audience, we explored *Aggregate* and *Event-level extrapolation* approaches, both using panel data as the source of information, which is then encoded as sets of Virtual People.

The *Aggregate* approach provides flexibility for TV modeling and allows for encoding arbitrary TV estimates into Virtual People, and therefore, it is compatible with any existing TV measurement methodology. If the TV modeling pipeline has no access to campaign-level digital measurement, then the overlap between TV and digital audiences is estimated via a conditional dependence statistical model. However, when digital measurement results can be routed to the TV modeling pipeline, the system can incorporate arbitrary correlations between the TV and digital audience, which could be learned, for instance, from panel data at the campaign level. Arbitrary correlation modeling techniques can be implemented, as long as they are using digital measurement as an immutable input.

The *Event-level extrapolation* assignment approach is novel for the measurement space and is yet to be deeply studied in practical circumstances. Its benefit is that is guarantees full consistency for the measurement of the TV audience, which is a desirable property for real-time estimation and optimization.

We have performed simulations [8], illustrating both the *Aggregate* and *Event-level extrapolation* approaches and showing that they produce results that closely replicate the underlying aggregate measurement model.

The set of Virtual People is large and a naive assignment algorithm of Virtual People to panelists is quadratic in runtime, which could have large computational costs. For the aggregate approach the assignment has to happen at the report request time, which is likely to be latency sensitive. We have presented a Deep LiquidLegions sketch data structure and an associated algorithm that runs over it and efficiently creates sketches of samples of Virtual People. This algorithm can be used for creating sketches of panelists, as well as for building the sketches for the aggregate approach. Simulations [7] confirm that the resulting sketches produce results that are very close to the guiding aggregate level model.

References

[1] P.A.M. Dirac, *The principles of quantum mechanics*, Comparative Pathobiology - Studies in the Postmodern Theory of Education, Clarendon Press, 1981.

[2] Shen fu Tsai, Evgeny Skvortsov, and Jim Koehler, *Hll-based tv panel audience extrapolation compatible with online audience measurement from logs (to appear)*, Tech. report, Google Inc, 2021.

[3] Jim Koehler, Evgeny Skvortsov, Sheng Ma, and Song Liu, *Measuring cross-device online audiences*, Tech. report, Google, Inc., 2016, available at `https://research.google/pubs/pub45353/`.

[4] Jim Koehler, Evgeny Skvortsov, and Wiesner Vos, *A method for measuring online audiences*, Tech. report, Google Inc, 2013, available at `https://research.google/pubs/pub41089/`.

[5] Benjamin Kreuter, Craig William Wright, Evgeny Sergeevich Skvortsov, Raimundo Mirisola, and Yao Wang, *Privacy-preserving secure cardinality and frequency estimation*, Tech. report, Google, LLC, 2020, available at `https://research.google/pubs/pub49177/`.

[6] Christian Schindelhauer and Gunnar Schomaker, *Weighted distributed hash tables*, Proceedings of the seventeenth annual ACM symposium on Parallelism in algorithms and architectures, 2005, pp. 218–227.

[7] Evgeny Skvortsov, *Deep LiquidLegions sampling*, (2020), available at `https://colab.research.google.com/github/world-federation-of-advertisers/virtual_people_examples/blob/main/notebooks/DeepLiquidSampling.ipynb`.

[8] Evgeny Skvortsov, *Modeling linear tv reach with virtual people*, (2020), available at `https://colab.research.google.com/github/world-federation-of-advertisers/virtual_people_examples/blob/main/notebooks/TV_modeling_with_Virtual_People.ipynb`.

[9] Evgeny Skvortsov and Jim Koehler, *Virtual people: Actionable reach modeling*, (2019), available at `https://research.google/pubs/pub48387/`.