Web Appendix

September 21, 2023

These materials have been supplied by the authors to aid in the understanding of their paper. The AMA is sharing these materials at the request of the authors.

Contents

A. Data Summaries

In this appendix, we provide a number of additional summaries for our data: first, we describe how the number of impressions of a video tends to grow, to support our claim that tracking a video over two weeks is sufficient to capture most of its growth. Then, we give additional descriptions of and summary statistics for the metadata of each TikTok post, and the metadata about the influencers (i.e., creators).

A.1. Impression Growth

Videos on TikTok tend to attract more attention when they are first posted. Then, as new videos and trends appear, their impression growth flattens out. In Figure A-1, we plot the impression growth curves for a random sample of videos for one month. We first find that, although some videos generate most of their impressions within the first few days, others can take longer. For all videos, impression growth slows substantially after two weeks. Therefore, in our analyses, we use the total number of impressions at the end of two weeks as our measure of a video's impressions. Two-week impressions capture more than 90% of a video's one-month impressions, while shorter durations (e.g., one week) capture less than 70% of a video's one-month views.

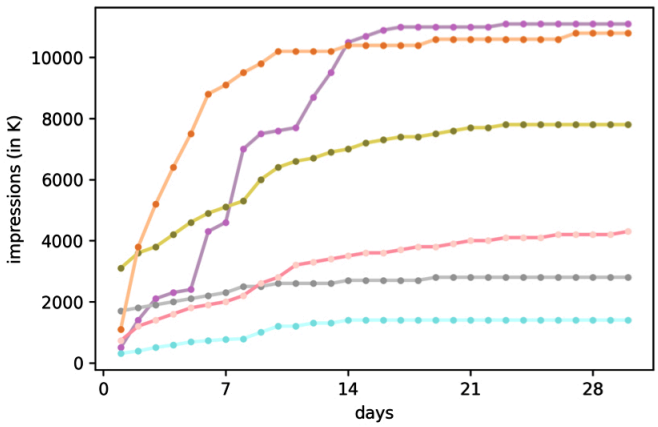


Figure A-1: Impression Growth Curves for a Random Sample of Videos

A.2. Post Statistics

For each post, our post statistics are metadata containing: 1) the hashtags used by the post, 2) the ranking and age of the post within the hashtag, and 3) the sponsorship status of the post. We now describe each of these variables in more detail:

- **Hashtags** Each post may contain multiple hashtags, which influence how widely a video is seen. One such hashtag is #fyp, which stands for "For You Page." The For You page (FYP) is TikTok's personalized landing page, where the platform recommends videos to its users based on its internal algorithm. It is widely believed that including #fyp on a post increases the likelihood of a post being featured on the FYP. Thus, we include an indicator variable to capture whether a post contains the #fyp hashtag ("Has FYP"). We also track the total number of hashtags used in a post ("Num hashtags") and among them how many are *trending* ("Num trending").

- **Rank and Age** On the Discover page, under each hashtag, there is a list of videos. Videos listed towards the top of the page get more exposure than those lower down the page. We track a video's rank under each hashtag on the Discover page ("Ranking"), and whether the hashtag was trending ("Is trending"). Finally, for each day that a post is tracked, we record how old the post is ("Video age") and how old the hashtag is ("Hashtag age") to represent their relative trendiness.

- **Sponsorship** For each post, we check if it is sponsored by a company ("Is sponsored") or organic.

Summary statistics for each of these variables are shown in Table A-1.

Table A-1: Summary Statistics of the Post Statistics

Variable	N	Mean	Std. Dev.	Min	Median	Max
Num Hashtags	518,303	6.662	3.275	0	6	47
Num Trending	518,303	1.078	0.853	0	1	10
Has FYP	518,303	0.604	0.489	0	1	1
Hashtag Age	518,303	6.003	13.316	0	1	148
Is Trending	518,303	0.767	0.423	0	1	1
Ranking	518,303	1029	586.495	1	1,037	2,000
Video Age	518,303	35.212	102.947	0	1.327	2,015
Is sponsored	518,303	0.001	0.031	0	0	1

A.3. Creator Statistics

For each post, we collect creator metadata from TikTok. These metadata capture basic statistics about the influencer, and are used in our model as controls. In a separate analysis not reported in the main text, we also attempted to gather information about creators from the posts themselves. The results in the paper were virtually identical, regardless of whether we included these features or not. Hence, they are not used in the main body of the paper, but we include them in this appendix for reference.

Creator Metadata Based on the creator metadata, we track how many accounts they are following (`Num_Following`), the total number of posts they have created in the past (`Num_Videos`) for each creator. We also check if they are verified users ("Is verified") as a general measure of their authorized speciality from various aspects. We include this feature because TikTok's algorithm might promote the content of verified users differently. Users could also be more likely to watch and share videos from verified users given their higher authenticity and specialty. Both could moderate the causal relationship between followers and video impressions and thus we include it as part of our covariates. We report summary statistics for this metadata in Table A-2.

Table A-2: Summaries of the creator metadata

Variable	N	Mean	Std. Dev.	Min	Median	Max
log(Following)	518,303	2.169	0.784	0	2.210	4.000
log(Num_Videos)	518,303	2.092	0.678	0	2.167	4.167
Is verified	518,303	0.003	0.055	0	0	1

Extracted Creator Characteritics The analyses in the main text use just the creator metadata described previously. As a robustness check, we also obtained more granular information about the creators themselves, by analyzing the faces in their videos. Specifically, we detected faces in the key image frames of each video and then extracted facial attributes including gender, age, ethnicity, and emotions from each frame via DeepFace, a pre-trained face recognition model (Taigman et al. 2014). In order to specify the face of the specific creator (since many videos have quite a few faces in them), we determined the most recurring face across the frames of each video and,

if possible, across videos posted by the same user ID. We summarize these features in Table A-3. When we included these features as controls in our Deep IV framework, the results did not change. Hence, these features are not included in any of our analyses. We include them in this appendix only for reference.

Table A-3: Summaries of the creator features extracted from the videos. Our results did not change when these features were used, and thus, they are not included in any of our analyses. We include them in this appendix only for reference.

Variable	N	Mean	Std. Dev.	Min	Median	Max
Gender						
Male	518,303	0.382	0.486	0	0	1
Female	518,303	0.618	0.486	0	1	1
Age Group						
10-19	518,303	0.375	0.484	0	0	1
20-29	518,303	0.466	0.499	0	0	1
30-39	518,303	0.132	0.338	0	0	1
40-49	518,303	0.018	0.133	0	0	1
50+	518,303	0.009	0.094	0	0	1
Race						
Asian	518,303	0.058	0.233	0	0	1
Black	518,303	0.265	0.441	0	0	1
Indian	518,303	0.054	0.226	0	0	1
Latino	518,303	0.351	0.477	0	0	1
Middle Eastern	518,303	0.023	0.150	0	0	1
White	518,303	0.249	0.432	0	0	1
Emotion						
Angry	518,303	0.127	0.332	0	0	1
Fear	518,303	0.009	0.094	0	0	1
Neutral	518,303	0.011	0.104	0	0	1
Sad	518,303	0.158	0.060	0	0	1
Disgust	518,303	0.214	0.410	0	0	1
Happy	518,303	0.343	0.474	0	0	1
Surprise	518,303	0.138	0.345	0	0	1

B. Technical Details: Feature Learning

B.1. Extracting Content Features

From each post on TikTok, we extract features from four modalities: (1) textual features, from both the video and the caption attached below; (2) image features, which are learned from the image frames of the video; (3) audio features, which capture features of the video's sound; and (4) editing features, which capture how creators edit their videos on the platform. While the first three are standard in video analysis, the fourth is a unique modality for TikTok. TikTok features numerous built-in editing tools for videos, which allow creators to add different effects to their videos (e.g. visual filters, musical and animal voice effects). Given the centrality of these features to TikTok posts, we create an innovative set of features to measure their presence in a post. We now describe each of these feature sets in more detail.

Text Our textual features are based on three sources: the video's caption, any text overlaid on the videos themselves (referred to as *stickers*), and text capturing the words spoken in any video voiceovers. Both the video description and the sticker text are readily extracted from the TikTok post. For voiceover, we first identify if the background music of a video belongs to any sound types related to speaking (e.g. speech, conversation, or monologue) using YAMNet, a pre-trained audio classification model (Gemmeke et al. 2017; Hershey et al. 2017). If so, we then use Google's speech-to-text API to convert the audio to words.[1] Across all three text modalities, we further process raw text into word embeddings, leveraging the pre-trained Word2Vec model (Mikolov et al. 2013). Hence, our final set of textual features is a collection of word embeddings across these three sources.

Image The visual component of the video data consists of sequences of image frames. In our analysis, rather than extracting and analyzing each individual image frame, of which there may be thousands, we extract image frames in five-second intervals. By doing so, we greatly reduce the volume of data, while still maintaining a relatively rich set of image frames.[2] We apply two pre-trained architectures to process these video frames into more meaningful features.

[1] https://cloud.google.com/speech-to-text

[2] A TikTok video is normally 3-60 seconds long with 30-60 frames per second.

The first architecture is the popular VGG-19 model, a 19-layered deep neural network originally trained to classify 1,000 unique classes of objects in images (Simonyan and Zisserman 2014). For each of our image frames, we feed the frame through VGG-19, and extract the second-to-last (4,096-dimensional) fully connected layer. Intuitively, the layers of VGG-19 learn representations of image content at increasing degrees of abstraction. Thus, the later layers of the model yield a high-level representation of the image content. A robust set of previous studies has shown that the final layers of the VGG-19 architecture can serve as the basis for a variety of other prediction tasks (e.g. Wei et al. 2019; Zha et al. 2015), exactly because they form an abstract representation of the image content, which is what we take advantage of here. In addition to the 4,096-dimensional representation, we also include as visual features the actual 1000-dimensional object probabilities returned by VGG-19. These probabilities capture whether one of the 1,000 classes of objects VGG-19 was trained on is present in a shot, giving us a more concrete understanding of what exactly was in each frame of the video.

The second architecture is a pre-trained convolutional neural network model trained on the SentiBank data (Borth et al. 2013). The SentiBank dataset is a set of tagged images used for "visual sentiment" analysis. The model provides probabilities over two-word (adjective-noun) image concepts, like "creepy house" or "happy dog." In contrast to simple sentiment schemes such as positive-negative, the SentiBank ontology is richer and captures many of the concepts actually used in social media posts. The number of visual concepts is quite large (2,089) and the output of the SentiBank is a vector of probabilities of 2,089 length.

In sum, for each image frame, our set of features consists of three vectors: the 4,096-dimensional VGG-19 representation, which captures a holistic representation of the image; (2) the 1,000-dimensional VGG-19 object class probabilities, which captures a wide range of potential objects that might be present in the image; and (3) the 2,089-dimensional SentiBank visual sentiment probability vector, which captures emotional features of the image. In the end, we performed a max-pooling over the image frames to generate a single object vector and visual sentiment vector for maximal inclusiveness. Together with the matrix of VGG-19 features with dimension #frames$\times$4096, they summarize all the visual characteristics extracted for each video.

Audio The other key aspect of the video data is the audio track. In fact, audio is a central component of TikTok, as it often provides a unifying link between videos within a hashtag. The background audio on TikTok can be selected from TikTok's music library, from music clips used in other videos, or uploaded as "original sound" by the creator. Our audio features try to capture both the features of the sound itself, as well as how the sound connects to other videos on the platform. Specifically, we collect three types of audio features:

- **Universality:** A single number from 0 to 1 that captures how many other videos within a hashtag use the same audio.
- **Sound classes:** We feed the audio through YAMNet (Gemmeke et al. 2017; Hershey et al. 2017), a deep neural network that predicts 521 audio event classes, based on the AudioSet-YouTube data. These features include various sound bits that characterize laughter, conversation, and different musical instruments. The specific feature we use is the 521-dimensional output probabilities, averaged over the entire sound track.
- **Acoustic features:** Finally, we extract standard acoustic features, capturing the raw characteristics of the sound itself (e.g., intensity and pitch).

In summary, the audio for video is summarized by a 543-dimensional representation.

Editing TikTok offers various tools that allow users to easily add special effects to their videos, including simple edits, such as video length and speed, and more complex edits, such as image filters. These tools are frequently used and are often highly salient in the final videos posted on the platform. We summarize these features below:

- **Video length:** The actual length of the video, in seconds, which tends to correlate with video content. For instance, cooking-related videos can be either short (e.g., just show the finished product), or long (e.g., tutorial-style with all the steps).
- **Stickers:** Whether the videos have textual stickers added to them, and if so, how many, and how many words are in each.
- **Scene length:** To quantify the overall speed of visual transitions of a video, we use the average scene length, which might vary remarkably from video to video. Specifically, we identify

the scene boundaries or transitions that occur in a video by identifying the discontinuity between two consecutive image frames. We adopt traditional intensity-based scene detection algorithms that compare the sharpness of changes in color histograms across consecutive frames. With the scene transitions, a video scene is identified as the content between each two consecutive transitions, and the length of a scene is then the length of the time period between the two transitions. Finally, we take the average of the scene lengths among all detected transitions.

- **Feature Variance:** Through editing, a creator could artificially create different content complexity to fit different video genres. For example, in the case of videos of news or story sharing with negligible background fluctuations, there are very small changes in both the spatial and temporal domains. On the other extreme, in the case of high-action content, such as DIY projects, they are normally accompanied by rapid visual changes throughout the video. In the paper, we define both the spatial and temporal content complexity of a video as the average variance of the object/sentiment probabilities across its key frames and across the object/sentiment classes. Mathematically, for any video i,

$$\text{Spatial Content Complexity}_{i,k} = \frac{1}{|\mathcal{T}_i|} \sum_{t \in \mathcal{T}_i} \left(\frac{1}{|\mathcal{S}_k|} \sum_{i \in \mathcal{S}_k} (p_{ijt} - \overline{p_{it}})^2 \right)$$

$$\text{Temporal Content Complexity}_{i,k} = \frac{1}{|\mathcal{S}_k|} \sum_{j \in \mathcal{S}_k} \left(\frac{1}{|\mathcal{T}_i|} \sum_{t \in \mathcal{T}_i} (p_{ijt} - \overline{p_{ij}})^2 \right)$$

  Here, j represents the class, $k \in \{\text{object, sentiment}\}$, $\mathcal{S}_k$ denotes the set of k, and $\mathcal{T}_i$ denotes the set of key image frames for video i. Then, p_{ijt} represents the probability of class j in frame t for video i, $\overline{p_{it}}$ represents the mean probability of all classes in frame t for video i, and $\overline{p_{ij}}$ represents the mean probability for class j across all image frames for video i.

- **Filters:** We cannot directly assess if a filter has been applied. However, we can measure the aesthetic quality of the image with metrics such as lightness and symmetry, which are affected by the use of image filters (Bhattacharya et al. 2013).

In total, we learn a 157-dimensional vector of editing features from the data.

B.2. Representation Learning Model

While the feature extraction process helps structure the raw data, the dimensionality of the feature space is still massive. To make these features more amenable for use within the deep IV framework, while also accounting for synergies and correlations across the different modalities, we build a representation learning model that embeds TikTok posts in a lower-dimensional space. Specifically, our framework is based on the variational autoencoder (VAE), a deep generative model used previously for learning representations of multimodal data (Dew, Ansari, and Toubia 2022). It takes as input all of the content-related features, and returns a vector representation $\boldsymbol{r}$ that captures the essence of the content, so that given $\boldsymbol{r}$, the original content could be reconstructed. Since our model is structured and captures features across multiple modalities, we call it the *structured, multimodal VAE*, or SMVAE. Before describing our SMVAE, we briefly describe variational autoencoders.

B.2.1. Variational Autoencoders

The variational autoencoder is a type of autoencoder, which is a machine learning model that has two key components: an encoder that compresses the data to a dense vector representation and a decoder that reconstructs the original data from that representation. The variational autoencoder is a probabilistic variant of this framework, where the generative process of the observed data, $\mathbf{x}_i$ for observation i, is modeled as a function of latent lower-dimensional vector representations, $\boldsymbol{r}_i$. Mirroring the classic autoencoder, the VAE also has two parts - the encoder, or the inference network, specifies a variational distribution over the latent space $q_\phi(\boldsymbol{r}|\boldsymbol{x})$ that approximates the true posterior distribution $p(\boldsymbol{r}|\boldsymbol{x})$ of the latent variables ($\boldsymbol{r}$) based on the observed data ($\boldsymbol{x}$), given a unit normal prior on $\boldsymbol{r}$. The decoder, or the generative network, models the generative process $x \sim p_\theta(\boldsymbol{x}|\boldsymbol{r})$ where $p_\theta(\boldsymbol{x}|\boldsymbol{r})$ is the probability distribution of the observed data ($\boldsymbol{x}$) given the latent variables ($\boldsymbol{r}$). Thus, $\boldsymbol{r}$ acts as a sufficient statistic for the data: given sufficiently rich encoders and decoders, $\boldsymbol{r}$ captures all of the information from the original data $\boldsymbol{x}$.

To learn representations of the data, we thus need to learn the amortized (i.e., shared across all observations) encoder and decoder parameters, ϕ and θ, respectively. We do so by minimizing the Kullback-Leibler (KL) divergence between the true posterior and the approximate posterior, as a

function of θ and ϕ. This minimization corresponds to maximizing the evidence lower bound, or ELBO, given by:

$$ELBO(\theta, \phi) = \mathbb{E}_{\boldsymbol{r} \sim q_\phi(\boldsymbol{r}|\boldsymbol{x})}[\log p_\theta(\boldsymbol{x}|\boldsymbol{r})] - D_{KL}[q_\phi(\boldsymbol{r}|\boldsymbol{x})||p(z)] \tag{A-1}$$

Here, $D_{KL}[\cdot||\cdot]$ is the KL divergence. The first term of the above equation can be interpreted as maximizing the reconstruction accuracy. The second term serves as a regularizer that penalizes estimates that are far from the prior. In other words, the log-likelihood encourages the decoder to reconstruct the data well, while the regularizer ensures that the encoder learns smooth latent representations of the input data. In the standard VAE model, the latent variables are not shared across data points, so the ELBO can easily decomposed into a sum where each term depends only on a single data point:

$$ELBO_i(\theta, \phi) = \mathbb{E}_{\boldsymbol{r}_i \sim q_\phi(\boldsymbol{r}_i|\boldsymbol{x}_i)}[\log p_\theta(\boldsymbol{x}_i|\boldsymbol{r}_i)] - D_{KL}[q_\phi(\boldsymbol{r}_i|\boldsymbol{x}_i)||p(r_i))] \tag{A-2}$$

This objective function can then be maximized using standard numerical methods.

B.2.2. SMVAE

We extend the previous VAE framework to accommodate the different modalities of TikTok posts. First, we use modality-specific encoders that capture the specifics of a given modality (e.g., audio). Next, we combine the modality-specific encoders using a neural network, which allows them to interact in a potentially nonlinear manner. This joint econder is then used to estimate the latent representation of a post. The decoder mirrors this structure. Given the highly structured nature of the modality-specific encoders and decoders, we refer to our proposed model as a structured, multimodal VAE, or SMVAE. We now describe each component in more detail.

Encoder Recall the goal of the encoder is to infer the approximate posterior distribution of the latent representation $\boldsymbol{r}_i$ for video i, given its content, $\mathbf{x}_i$. In line with the typical assumption employed in a standard VAE, we assume a Gaussian approximation to the posterior, such that

$$p(\boldsymbol{r}_i \mid x_i) \approx q(\boldsymbol{r}_i \mid \boldsymbol{\mu}_i, \boldsymbol{\sigma}_i I). \tag{A-3}$$

Thus, the encoder maps the data $\mathbf{x}_i$ to the variational parameters $\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i$. The encoder has two parts: first, each modality is encoded separately through a modality-specific encoder. We index modalities by $m = 1, \ldots, M$, and denote this first-stage modality specific encoding as:

$$\mathbf{h}_i^m = \text{ModalityEncoder}_{\phi_m}^m(\mathbf{x}_{im}), \tag{A-4}$$

where $\mathbf{h}_i^m$ is the intermediate representation learned from video i's modality m. Typically, these modality-specific encoders take the form of a deep neural network, parameterized by ϕ_m, whose architecture varies depending on the modality. Subsequently, the intermediate representations are concatenated together using a joint encoder, denoted by,

$$(\boldsymbol{\mu}_i, \log \boldsymbol{\sigma}_i) = \text{JointEncoder}_{\phi_0}([\mathbf{h}_i^1, \ldots, \mathbf{h}_i^M]), \tag{A-5}$$

where the function is again modeled using a deep neural network, parameterized by ϕ_0.

For each modality, we structure its encoder to reflect the unique structure of the data of that modality. We outline each of these architectures here:

- **Text:** The input to the text encoder is the sequential list of word embeddings from each of the three sources of text (video description, sticker text, and voice-over). To extract features from each of these word embeddings, we pass them separately through a Bi-LSTM layer, which is the typical choice to accommodate sequences of text, as it learns the meaning of a sentence by processing word sequences in both forward and backward directions (Schuster and Paliwal 1997). The output of the Bi-LSTM is a vector of hidden states, which constitute the text-specific representation, $\mathbf{h}_i^{\text{Text}}$.

- **Image:** The input to the image encoder includes the vectors of VGG-19 features for each of the video frames, the 1000-D object vector, and the 2089-D visual sentiment vector. To extract features from the sequential VGG-19 features, we pass them through a LSTM layer that combines them into a single 4096-D VGG-19 vector while keeping the temporal relations between each frame. We then concatenate the hidden states of this LSTM with the other two input modalities, and pass it through several fully connected layers, to learn the video-specific representation, $\mathbf{h}_i^{\text{Image}}$.

- **Audio:** The input to the audio encoder is the 543-D audio feature vector, including the

percentage measure of music universality, the music classes, and the ten standard acoustic features. We pass these through through multiple fully connected layers to get the audio-specific representation, $\mathbf{h}_i^{\text{Audio}}$.

- **Editing:** The input to the editing encoder is the 157-D editing feature vector, which we pass through a single fully connected layer to get the editing-specific representation, $\mathbf{h}_i^{\text{Editing}}$.

- **Joint:** Finally, the joint encoder takes each of the modality-specific representations, $\mathbf{h}_i^{\text{Text}}$, $\mathbf{h}_i^{\text{Image}}$, $\mathbf{h}_i^{\text{Audio}}$, and $\mathbf{h}_i^{\text{Editing}}$, and feeds them through a single-layered fully connected network to provide the variational parameters $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i$.

Decoder The goal of the decoder is to reconstruct the observed data from the multimodal representation, $\boldsymbol{r}_i$. This reconstruction involves specifying a generative model for the data $\mathbf{x}_i$, parameterized as a function of the representation $\boldsymbol{r}_i$, which itself is given a unit normal prior. We assume that, conditional on $\boldsymbol{r}_i$, each of the modalities is independent, and thus can be specified using a modality-specific decoder. Mirroring our notation for the encoder, we generically write,

$$\hat{\mathbf{x}}_i^m = \text{Decoder}_{\theta_m}^m(\boldsymbol{r}_i),$$

where $\hat{\mathbf{x}}_i^m$ indicates the parameters governing the data generating process for $\mathbf{x}_i^m$. For instance, for words, $\hat{\mathbf{x}}_i^m$ is the probability of seeing that word. For real-valued features, $\hat{\mathbf{x}}_i^m$ is the expectation of that value. The specific architectures for each modality's decoder are:

- **Text:** To generate text, the latent representation is first passed through a fully connected layer to create separate Bi-LSTM inputs for the three sources of text: video description, sticker text and voice-over. Then for each of them, we pass the Bi-LSTM outputs through a time distributed fully connected layer with softmax activation to get the probability of each word at each given time step.[3] Therefore, for each video, the output of the text decoder is a matrix of probabilities for every word and each position in the text. We employ categorical cross-entropy loss for the text reconstruction, which corresponds to modeling the text with a categorical likelihood.

[3]The time distributed layer applies the same activation function to each of the time stamps.

- **Image:** The image decoder reconstructs the VGG-19 feature matrix, the object and the visual sentiment vector. To do so, we first pass $\boldsymbol{r}_i$ through multiple fully connected layers to obtain (1) the inputs for an LSTM, which is then used to reconstruct the VGG-19 features; and (2) the reconstructed object and visual sentiment vector means. We employ a mean squared error loss for the image reconstruction, which corresponds to a Gaussian likelihood for the image features.

- **Audio and Editing:** Both of these modalities use a single-layered fully connected network as decoders, together with the mean squared error loss for feature reconstruction, again corresponding to a Gaussian likelihood over the features.

Overall Loss To optimize the parameters θ and ϕ, we use the same ELBO loss as defined for the VAE in Equation A-2, but extend it to capture the reconstruction losses for the multiple domains. Specifically, the loss for our SMVAE model is as follows, where for simplicity, we suppress the notation of domain-specific parameters θ and ϕ in the right hand side parentheses and the four modalities $\{\text{Text}, \text{Image}, \text{Audio}, \text{Editing}\}$ by $\{t, i, a, e\}$, respectively:

$$\mathcal{L}_{\text{SMVAE}}(\theta, \phi) = \sum_{m \in \{t,i,a,e\}} \mathcal{L}_{\text{recons},m} = \mathcal{L}_{\text{recons}_t} + \mathcal{L}_{\text{recons}_i} + \mathcal{L}_{\text{recons}_a} + \mathcal{L}_{\text{recons}_e} + \mathcal{L}_{KL}(\theta, \phi). \quad \text{(A-6)}$$

To minimize this loss, we use the Adam optimizer (Kingma and Ba 2014). The model is trained on batch size 32 for 300 epochs with early stopping and a learning rate of 1e-5. To prevent overfitting, we use an L2-regularizer on the weights of our model.

B.2.3. Architecture Details

The text encoder consists of a Bi-LSTM layer with dimension size 64 for the three text sources and a fully connected layer of the same size 64 for the concatenated output from the three Bi-LSTM layers. The image encoder consists of one LSTM layer with dimension size 4096 for the sequential VGG-19 features and three fully connected layers of sizes 4096, 1024 and 512 for the final image vector including the LSTM output of the VGG-19 features, the object vector and the visual sentiment vector. The audio and editing encoders consist of one fully connected layer of size 64 and 16, respectively. The joint encoder takes the concatenation of the encoded features of

size 656 from the four separate encoders as input and further compresses it into size 256 through one fully connected layer. The decoders consists of layers with the same dimensions yet inverted as those of the encoders. For all fully connected layers in both encoders and decoders, we use the Rectified Linear Unit (ReLU) as the nonlinear activation function.

We experimented with batch sizes 16, 32, 64, 128, 256, and found 32 to be optimal for the training of our SMVAE. The model was trained for 300 epochs with early stopping and a learning rate of 1e-5. To prevent overfitting, we use L2-regularization on the weights of our model. We experimented with weight penalties of 0, 0.05, 0.1, 0.3, and 0.6 and found 0.05 to be optimal for both the encoder and decoder. To minimize our SMVAE loss, we use Adam as the optimizer. As mentioned in the data section, our video data have around 1500-2000 videos under each of the hashtags. In total, we have more than 0.5M unique videos spanning over all hashtags. They are randomly split by hashtag into three parts: the training set, the validation set and the test set, at the ratio of 8:1:1. In other words, they are split randomly but the ratio is preserved within each hashtag. This strategy ensures, at least from the data perspective, that all impactful, recurring content attributes resulting from belonging to a specific hashtag on video impression growth can be well-captured, even under some imbalance in the video volume across hashtags.

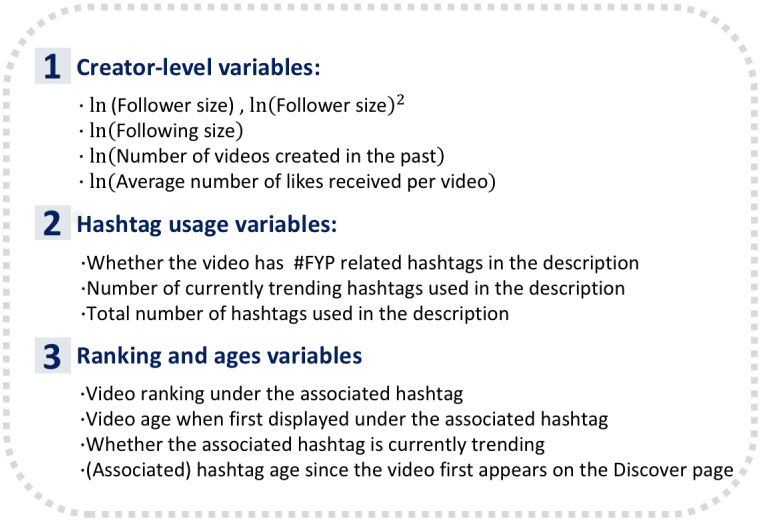
C. Validating the Representations

In the main body of the paper, we validate our representations in two ways: first, we look at the reconstruction accuracy from the SMVAE. Second, we see whether the representations can do well at predicting video popularity. In this appendix, we provide additional details for these two analyses. We start by giving the technical details of the popularity prediction task. Then, we illustrate how performance in both reconstruction and popularity performance degrades as modalities are removed, highlighting the importance of learning the joint multimodal representation. Finally, we report the results of a study not reported in the main text, where we illustrate that distance in representation space corresponds to meaningful differences between videos, further validating the representations.

C.1. Predicting Video Popularity

To illustrate that our representations can capture "good" vs. "bad" videos, we show that the representations can be used to predict video popularity. We proceed in two steps: the first part of our analysis aims to quantify general impression growth patterns, using only the non-content factors listed below. We refer to this step as "Stage 1" of our modeling framework. The results (described in more detail below) show that content unrelated features explain only a small part of the variance in the growth of impressions. We then leverage the content of videos as expressed by our SMVAE representations to assess. This step is what we refer to as "Stage 2" of our predictive model. The good performance of our "Stage 2" prediction indicates that our representations can well capture the quality differences across the video content that drives its surprisingly good or bad popularity growth.

Figure A-2: Non-content Factors



C.1.1. Stage 1: Baseline Model

We use a logarithmic growth specification as our baseline model, with the content-unrelated features as covariates. Intuitively, videos tend to have an initial period of fast impression growth, followed by relatively slow growth, which mirrors the assumptions of logarithmic growth. We estimate a Bayesian, multi-level logarithmic growth model, with post- and hashtag-level parameters, which are themselves predicted by the set of content-unrelated features. Let y_{ijt} denote the impression count for video j under hashtag i observed on day t, where $t = 1$ corresponds to the video's first appearance under the hashtag.[4] We specify y_{ijt} as follows:

$$y_{ijt} = A_{ij} \log(t) + B_{ij} + \varepsilon_{ijt} \tag{A-7}$$

$$A_{ij} = \exp(a_{ij}),\ B_{ij} = \exp(b_{ij}) \tag{A-8}$$

$$\varepsilon_{ijt} \sim \mathcal{N}(0, \tau^2) \tag{A-9}$$

where a_{ij} is the log growth rate of video j under hashtag i and b_{ij} is its log starting impression. To capture the variation in growth rates and starting impressions, we assume a hierarchical, multi-

[4] If a video has been listed under multiple hashtags, we only consider its earliest appearance to avoid duplication.

level structure on a_{ij} and b_{ij}, such that:

$$a_{ij} \sim \mathcal{N}(\alpha_{0i} + [\boldsymbol{x}_{ij}^{u}, y_{ij1}]'\boldsymbol{\alpha}_i, \sigma_{\alpha,i}^2), \tag{A-10}$$

$$b_{ij} \sim \mathcal{N}(\beta_{0i} + \boldsymbol{x}_{ij}^{u\prime}\boldsymbol{\beta}_i, \sigma_{\beta,i}^2), \tag{A-11}$$

where $\boldsymbol{x}_{ij}^{u}$ is the set of non-content covariates for video j under hashtag i. In short, our multi-level formulation assumes that each video's growth parameters arise from a combination of hashtag-specific effects $(\boldsymbol{\alpha}_i, \boldsymbol{\beta}_i)$ and the content-unrelated features for that video. The realized initial impression y_{ij1} is also included as an independent variable in the model of a_{ij}. Finally, each of these hashtag-level parameters is drawn independently from Gaussian priors:

$$\boldsymbol{\alpha}_i \sim \mathcal{N}(\boldsymbol{\mu}_\alpha, \sigma_\alpha^2 I), \tag{A-12}$$

$$\boldsymbol{\beta}_i \sim \mathcal{N}(\boldsymbol{\mu}_\beta, \sigma_\beta^2 I), \tag{A-13}$$

Since we don't have any prior knowledge, we specify diffuse distributions for the set of hyperparameters $\tau^2, \sigma_{\alpha,i}^2, \sigma_{\beta,i}^2, \mu_{\alpha_0}, \mu_{\beta_0}, \sigma_{\alpha,0}^2, \sigma_{\beta,0}^2, \boldsymbol{\mu}_\alpha, \boldsymbol{\mu}_\beta, \boldsymbol{\sigma}_\alpha^2, \boldsymbol{\sigma}_\beta^2$. More specifically, we assume that the parameters for the intercepts (μ_{α_0}, μ_{β_0}) and covariates ($\boldsymbol{\mu}_\alpha$, $\boldsymbol{\mu}_\beta$) are mutually independent. We assume a diffuse normal prior for both μ_{α_0} and μ_{β_0} and each element of $\boldsymbol{\mu}_\alpha$ and $\boldsymbol{\mu}_\beta$. In case of the variance terms, including both the estimation variance terms $\tau^2, \sigma_{\alpha,i}^2$ and $\sigma_{\beta,i}^2$ and the coefficient variance terms $\sigma_{\alpha,i}^2, \sigma_{\beta,i}^2, \sigma_{\alpha,0}^2, \sigma_{\beta,0}^2, \boldsymbol{\sigma}_\alpha^2$, and $\boldsymbol{\sigma}_\beta^2$, we follow the convention of specifying a gamma prior for the inverse of the variance of each parameter.

Crucially, using Bayesian R^2 (Gelman et al. 2019), we find that non-content covariates are able to explain only about 33% of the variation in a_{ij} and b_{ij} across all hashtags. That nearly two-thirds of the variation is unexplained suggests that a large number of videos may exceed, or fall short of, their "destiny" — the performance we would expect by looking at who posted the video, and under what hashtag. In the following subsection, we use the *content* of the videos for demand prediction.

C.1.2. Stage 2: Residual Prediction based on the Modeled Video Content

We build neural network predictors to predict both the direction and value of the unexplained impression growth from Stage 1 based on the learned video representations from our SMVAE

framework.

To measure how much the realized popularity growth of a video is different from its estimated baseline in the previous subsection, we calculate its growth residuals defined as follows

$$a_{ij} - \hat{a}_{ij} \ , \ b_{ij} - \hat{b}_{ij} \tag{A-14}$$

where the parameter a_{ij} (b_{ij}) represents the logarithmic fitted growth rate (starting impression) we ultimately try to predict. The estimate $\hat{a}_{ij}$ (or $\hat{b}_{ij}$) represents the estimated value determined by only the non-content baseline factors from the first-stage estimation. The difference then captures a video's growth rate (or starting impression) that cannot be explained prior to accounting for its content. A positive difference refers to over-performance, meaning that its high-quality content helps it to realize a growth rate (or starting impression) higher than expected (hidden gem). A negative difference, on the other hand, refers to under-performance as a result of the video's lower-than-expected popularity growth (lemon) under its unsatisfactory content. We refer to the former as surprisingly popular videos, and the latter as surprisingly unpopular videos.

Mathematically, the difference takes the following form:

$$a_{ij} - (\mu_{\alpha_0} + \boldsymbol{X}_{ij}\boldsymbol{\mu}_{\alpha}) \tag{A-15}$$

$$b_{ij} - (\mu_{\beta_0} + \boldsymbol{X}_{ij}\boldsymbol{\mu}_{\beta}) \tag{A-16}$$

It is calculated based on the common parameters ($\alpha_0, \boldsymbol{\mu}_{\alpha}, \beta_0$ and $\boldsymbol{\mu}_{\beta}$) representing all hashtags in our dataset, thus capturing not only the video-level but also the recurring content attributes resulting from belonging to a specific hashtag that help a video achieve its observed growth.

We use neural networks to flexibly capture the link between the video content and the unexplained impression growth. To classify over- and under-performing videos and predict the magnitude of their unexpected popularity (or lack thereof), we build sign classifiers and numeric predictors that take the learned video representations as input and provide the binary signs and the numeric values of the growth residuals, respectively, as an output. They consist of three fully connected layers with size 128, 64 and 1. We adopt the Rectified Linear Unit (ReLU) as the nonlinear activation function for the first two layers and softmax activation for the last layer of the sign classifiers and linear activation for that of the numeric predictors. They are trained by optimizing

the binary cross-entropy and mean squared error loss, respectively.

C.2. Reconstruction and Prediction Across Submodels

We have shown that our SMVAE modeling framework learns meaningful representations of the videos that effectively capture their content, with regard to both the subject of the post (i.e., what it's about), and the post's quality. However, different video components might contribute differently to its learning performance. In addition, the synergy across different components might also play an important role in capturing the overall video content. In this appendix, we compare different VAE models that learns videos representations from different (sets of) modalities based on the two validation criteria we have examined for our SMVAE model.

C.2.1. Out-of-sample Reconstruction Statistics

We start by showing how well we can reconstruct the original video content based on the latent representations learned from different VAE models with increasing complexity (i.e. more modalities). Specifically, we reconstruct our video input from each modality based on the learned representations. For text, we compare how many of the reconstructed words match to the original words at their respective positions. For the other three modalities, we calculate the mean squared error between the reconstructed and original features. For image modality, in particular, we need to average it over all image frames. The out-of-sample reconstruction statistics are provided in Table A-4. Overall, we see that more modalities provide complementary information, thus jointly improving the learning of the multi-modal video content, which in turn boosts the reconstruction performance. In particular, reconstruction based on the four separate single-modality models is much worse than that based on our SMVAE model with all four modalities, suggesting that there is indeed synergy across modalities captured by our model that helps characterize content for each modality.

Table A-4: Reconstruction Statistics under VAE Models with Different Modalities

	Text Accuracy	(Avg.) Image Loss	Audio Loss	Editing Loss
VAE(T)	0.14			
VAE(I)		21.42		
VAE(A)			2.61	
VAE(E)				0.83
VAE(T+I)	0.29	15.70		
VAE(T+I+A)	0.33	13.82	1.35	
VAE(T+I+A+E)	**0.38**	**10.81**	**0.97**	**0.69**

Note: For notation conciseness, we suppress the text, image, audio and editing modalities by "T", "I", "A", and "E" as shown in the parentheses.

C.2.2. Popularity Prediction Statistics

We now compare how well each of the nested submodels predicts the surprisingly high or low popularity of a video. We follow the same two-stage procedure as before, where, in the first stage, we account for standard predictors of impression growth, like who created the video and how many followers that person had, through a logarithmic growth model, exactly as in Web Appendix C.1. However, now, instead of using the full video representations in the second stage, we use the content representations learned from different VAE models as above.

In the Table A-5, we compare performance using two metrics. The first is prediction accuracy of classifying surprisingly popular (or unpopular) videos. We find that among the single modality models, the visual model that takes only the visual features as input performs better than the textual, audio, and editing models. In addition, although visual features perform better than the other input features, single modality models perform worse than the multimodal models. When we gradually extend our input data from just text to all four modalities, we observe improvement in the prediction results of unequal magnitude, with the largest one coming from the image modality followed by the editing modality. Our proposed SMVAE model with all four modalities outperforms the one without the editing modality by a huge margin for both parameters. More specifically, it increases the accuracy from 72.1% to 77.4% and from 65.6% to 70.4% for the growth rate and the starting impression, respectively. The second performance metric we consider

Table A-5: Binary and Numeric Prediction Results under Different Models

	Growth Rate		Starting Impression	
	Accuracy	MSE	Accuracy	MSE
VAE-based models				
VAE(T)	0.568	32.3	0.570	29.6
VAE(I)	0.653	28.0	0.602	25.9
VAE(A)	0.540	32.8	0.511	30.1
VAE(E)	0.525	31.3	0.509	28.8
VAE(T+I)	0.692	23.2	0.625	24.3
VAE(T+I+A)	0.721	22.4	0.656	22.3
VAE(T+I+A+E)	**0.774**	**18.5**	**0.704**	**15.6**

Note: For notation conciseness, we suppress the text, image, audio and editing modalities by "T", "I", "A", and "E" as shown in the parentheses.

is Mean Square Error (MSE) for predicting the exact values of the unexpected impression growth of a video and see similar patterns. Again, the visual model beats the textual, audio and editing models in single modality models. The SMVAE achieves the highest predictive power enhanced the most by the image and editing modalities, while the latter only significantly contributes to the prediction performance when accompanying other modalities.

Overall, the performance of our proposed SMVAE using all text, image, audio and editing features is superior to the other simpler VAE models. Together, these results suggest the importance of capturing all modalities together. The nonlinear gains from adding modalities also implies that the modalities work in synergy to explain the content and quality of videos.

C.3. Additional Validation of the Representations

In this section, we analyze the $\boldsymbol{r}$ representations themselves, to understand if different $\boldsymbol{r}$s map onto meaningful differences between videos. While the dimensions of $\boldsymbol{r}$ are not interpretable, distances between videos in $\boldsymbol{r}$-space are: we find that posts with similar values of $\boldsymbol{r}$ have similar features. We illustrate this similarity in Figure A-3, where we show the top three nearest neighbors in the latent $\boldsymbol{r}$ space for a set of three focal videos under three randomly selected hashtags. For each video, we show five frames, as well as the hashtag that the video was posted under. Overall, we see there is a high degree of similarity among neighboring videos. For example, the focal and neighboring videos under #FoodTikTok all start with a dish, followed by the cooking process, and then show the dish again. The color palettes across the frames are similar as well. Across all focal videos,

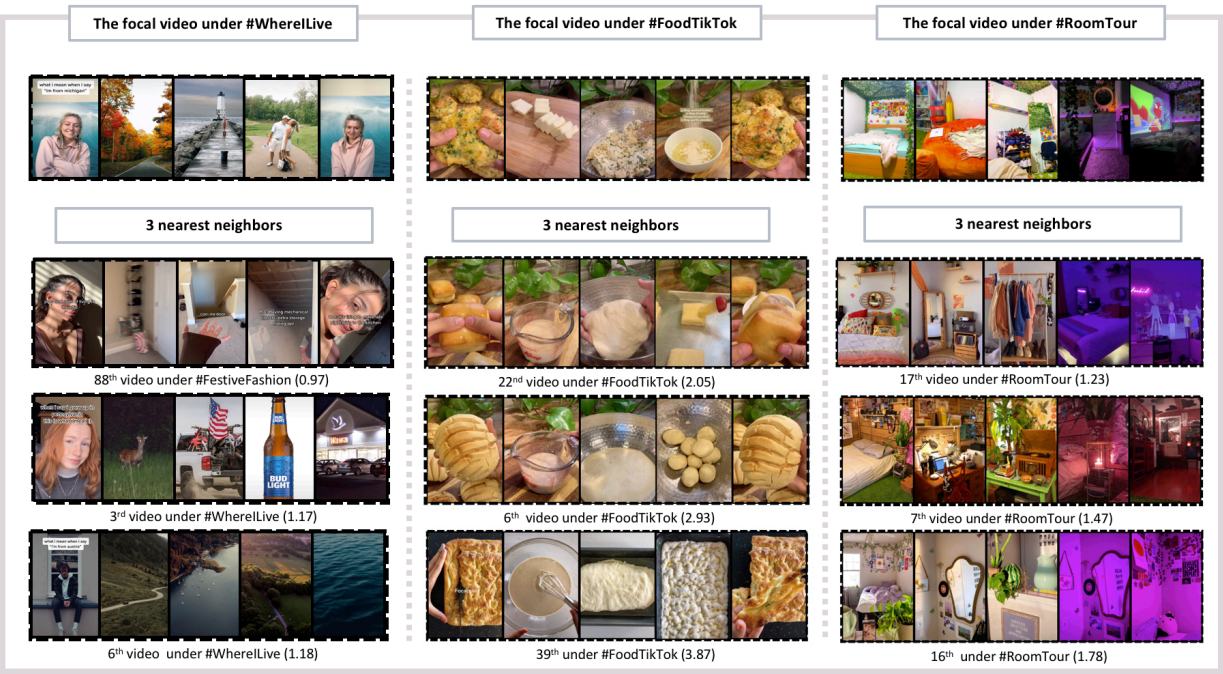


Figure A-3: Nearest Neighbors Case Studies
The three nearest neighbors in r-space for three randomly selected focal videos, selected from different hashtags.

neighboring videos follow a similar trajectory, color tones, and objects presented throughout. Less obvious from Figure A-3, neighboring videos share many other features, including the music used, the extent to which stickers, or overlaid textual comments, are used, and the scene length. In sum, this again suggests that r meaningfully captures the content of posts: videos that have similar rs have similar content.

D. Deep IV

In this appendix, we provide a summary of the Deep IV framework and showcase its relevance for addressing our research questions. We then provide details about our specific implementation, namely the architectures used for each of the neural network components.

D.1. Deriving Deep IV

We begin by describing Deep IV in more detail. We closely follow the exposition in Hartford et al. (2017). Our notation closely mirrors the notation from the main body of the paper.

Deep IV assumes the following model for the data generating process:

$$Y = g(F, X) + e, \tag{A-17}$$

where Y is the outcome variable; F is the treatment variable; X is a set of observed features that affect both F and Y; e are some unobserved variables, i.e., the error, that may be correlated with X, F, and Y; and $g(\cdot)$ is a flexible function of both X and F that captures the relationship of interest. In words, this model assumes a potentially nonlinear relationship between the treatment, controls, and the dependent variable (i.e., $g(\cdot)$), while assuming that any unobserved variables are additive (i.e., e). In our context, Y is the number of impressions a video receives, F is the number of followers the post's creator has, and X contains both the content features of the video ($\boldsymbol{r}$) and any other observables about the post and its creator that are not included in $\boldsymbol{r}$.

Our goal is to predict counterfactual impressions of a video (Y) under different follower size (F), conditional on other observables (X). To do so, Hartford et al. (2017) define a counterfactual prediction function,

$$h(F, X) \equiv g(F, X) + \mathbb{E}[e \mid X], \tag{A-18}$$

where, crucially, $h(\cdot)$ is defined such that the expectation of e does not depend on F. This definition means the expectation of the unobservables will not change depending on F, the follower size (i.e., h is a "structural" equation). In turn, this specification allows us to infer counterfactual values of impressions: given two treatment values (i.e., two different follower sizes), $h(F_1, X) - h(F_2, X) = g(F_1, X) - g(F_2, X)$, suggesting that, if we are able to estimate $h(F, X)$, we can infer these counterfactual quantities. However, estimating $h(F, X)$ is not straightforward. Naively,

we might imagine that we could estimate $h(F, X)$ by simply fitting a flexible model to predict Y from F and X. In our context, that would mean estimating a purely predictive model of impressions from follower count, creator observables, and video characteristics.[5] However, if there are unobservables that could be driving both Y and F (like, for example, the cross-posting behavior described previously), that predictive model will not correspond to $h(F, X)$: under Equation A-17,

$$\mathbb{E}[Y \mid F, X] = g(F, X) + \mathbb{E}[e \mid F, X] \neq h(F, X),$$

as we have allowed $\mathbb{E}[e \mid F, X] \neq \mathbb{E}[e \mid X]$. In simpler terms, if there is any variable we do not control for that can drive both followers and impressions, it will make the error term not independent of F. In turn, we will not be able to accurately infer counterfactual outcomes. Only in the case where there are no unobserved confounders will the predictive model allow us to estimate the true counterfactual prediction.

To get an unbiased estimate of $h(\cdot)$, even in the presence of possible unobserved confounders, Hartford et al. (2017) suggest using an instrument. Specifically, as in the standard instrumental variables setting, suppose there exists a variable Z, such that Z satisfies the following three conditions:

1. **Relevance:** $P(f \mid X, Z)$, the distribution of F given X and Z, is not constant in Z;
2. **Exclusion:** Z does not enter Equation A-17;
3. **Exogeneity:** Z is conditionally independent of the error, e;

then we can leverage this instrumental variable to estimate $h(\cdot)$. Specifically, given Z, we can write:

$$\mathbb{E}[Y \mid X, Z] = \mathbb{E}[g(F, X) \mid X, Z] + \mathbb{E}[e \mid Z] = \int h(F, X) dP(F \mid X, Z), \tag{A-19}$$

which allows us, theoretically, to estimate $h(F, Z)$ given two observable quantities: $\mathbb{E}[Y \mid X, Z]$ and $P(F \mid X, Z)$.

Until now, the derivation has closely followed the standard derivation of instrumental variables approaches (e.g., Newey and Powell 2003). The Deep IV framework departs from standard

[5] We also report the out-of-sample prediction results both for this purely predictive model and our causal model. As expected, direct prediction yields lower MSE. However, our causal model also performs reasonably well with relatively low MSE. See more details in Web Appendix E

approaches by operationalizing Equation A-19 directly. Hartford et al. (2017) propose directly optimizing an estimator $\hat{h}(\cdot)$ for $h(\cdot)$ by minimizing the objective function,

$$\min_{\hat{h}\in\mathcal{H}} \sum_{n=1}^{N} \left(Y_n - \int \hat{h}(F, X_n) dP(F|X_n, Z_n) \right)^2, \tag{A-20}$$

where $\mathcal{H}$ is defined as a set of functions that can be parameterized by a neural network, allowing $\hat{h}$ to be rewritten as $\hat{h} = h_\zeta(F, X)$, where ζ are the parameters of the neural network. Unfortunately, without knowing $P(F \mid X, Z)$, this equation cannot be used directly. Hence, Hartford et al. (2017) propose first estimating $P(\cdot)$ with a parametric distribution function $\hat{P}(f \mid X, Z)$. To allow maximal flexibility in the form of $\hat{P}$, Hartford et al. (2017) again parameterize it using neural networks, specifically assuming, for the case of a continuous treatment variable like followers, F, that $\hat{P}(F \mid X, Z)$ is the distribution function of a mixture of Gaussians, whose parameters depend on the data through a neural network with parameters η, subsequently denoted as $\hat{P} = P_\eta(F \mid X, Z)$. Thus, the Deep IV approach replaces the usual first stage of the two-stage least squares procedure by computing a flexible estimate $P_\eta(f \mid X, Z)$ of the distribution of the treatment given observables. Then, conditional on the learned η from stage one, denoted η^*, Deep IV solves

$$\zeta^* = \arg\min_{\zeta} \frac{1}{N} \sum_{n=1}^{N} \left(Y_n - \int h_\zeta(F, X_n) dP_{\eta^*}(F|X_n, Z_n) \right)^2 \tag{A-21}$$

to learn an estimate of $\hat{h}(\cdot) = h_{\zeta^*}(\cdot)$, which can be used to quantify counterfactuals of interest. In practice, both of these models (for h and P) are trained using stochastic gradient descent, and we refer readers to Hartford et al. (2017) for additional implementation details.

D.2. Implementation Details

As in Hartford et al. (2017), we train the networks in both stages of Deep IV via stochastic gradient descent. To select the best hyperparameters, we perform out-of-sample validation for both stages by simply evaluating their respective losses on held out data. Each stage is evaluated sequentially, with the second stage validation based on the best possible network from the first stage. In the end, for the model for F we find the optimal architecture is a mixture of 10 Gaussian distributions, with 3 fully connected layers with sizes 128, 64 and 32 for the network modeling their parameters. For the model of h, we find an optimal structure of 4 fully connected layers, with size 128, 64,

32 and 1, respectively. For all layers, we use the Rectified Linear Unit (ReLU) as the activation function. The model is trained on batches of size 256 for 100 epochs with early stopping and a learning rate of 1e-5. To prevent overfitting, we set the dropout rate to be 0.2.

E. Out-of-sample Prediction

While our model is a causal, rather than predictive, model, which uses only the plausibly exogenous variation in followers to predict the number of impressions of a video, we can still use it to make predictions. In this section, we evaluate it as a predictive model. Specifically, we test how well the model is able to predict the impressions of videos in a holdout sample, and compare that performance to several modern benchmarks. In particular, we consider three benchmarks:

1. A neural network *without* using the instrumental variable, with an architecture exactly mirroring our Deep IV architecture, but without the first stage.

2. A random forest model, which uses all of the same inputs as our Deep IV model, and is trained to predict impressions, again, leveraging the full variation in followers (i.e., no IV).

3. Two-stage least squares (2SLS), i.e., the standard IV linear model, using the same IV as our Deep IV framework.

In Table A-6, we compare the performance of these models, using the (out-of-sample) Mean Square Error (MSE). We find that, unsurprisingly, direct prediction (i.e., no IV) yields lower MSE than prediction using only part of the variation in follower sizes (i.e., IV-based methods). Yet still, our causal model performs reasonably well, achieving an almost identical MSE to the random forest model. Moreover, the linear 2SLS's MSE is over 100% worse than our deep IV specification, further emphasizing the importance of considering nonlinearities.

Table A-6: Out-of-sample Prediction Results
Out-of-sample prediction results for our model, Deep IV (in bold), versus three benchmark models. The outcome is the mean squared error (MSE) of predicting the number of impressions for a held-out video. Despite using only plausibly exogenous variation, our specification performs competitively with purely predictive frameworks like the random forest.

Model	MSE
Neural Network, No IV	2.65
Random Forest, No IV	6.06
Deep IV	**6.10**
2SLS	12.48

F. Deriving the Hashtag Topics

Our classification of hashtags into topics involves two steps: first, we represent each hashtag in a vector space, by learning embeddings of the words within each hashtag, then combining the word embeddings together. Then, given these hashtag embeddings, we simply cluster them to learn the hashtag topics. Hashtags are typically concatenations of common words, like "#HomeOffice" merges "home" and "office." Hence, to learn representations of the words in a hashtag, we first need to split the hashtag into its actual word components. We do that by pattern matching, subsetting the overall hashtag name into strings which can be matched to a dictionary. Then, the words that are found are matched to pre-trained GloVe embeddings. This yields 50-dimensional word embeddings for each segmented word. To combine them into a single embedding, we use average pooling over all embeddings in the hashtag name. As an example, consider the hashtag "#HomeOffice": we first segment it into "Home" and "Office" and take the average of the two word embeddings as the semantic representation for "#HomeOffice". Consequently, each hashtag is represented by a single 50-dimensional vector. To classify these hashtags into topics, we use simple K-means clustering, wherein we select the number of topics using the standard elbow method. This procedure produces five clusters of hashtags, which we call our topics. To assign labels to these topics, we cross-reference the hashtags within each cluster to a list of popular content categories from TikTok.[6] The resulting topics are: 1) life, 2) holidays, 3) skills, 4) food, and 5) gaming.

[6] https://www.statista.com/statistics/1130988/most-popular-categories-tiktok-worldwide-hashtag-views/

G. Sponsored vs. Organic Hashtags

Sponsored or campaign challenges are hashtags created by firms and paid to be promoted on TikTok. Similar to an organic hashtag, a sponsored hashtag also consists of a banner displayed on the Discover page and a main page that contains a description of what users need to do in order to participate in the challenge and promote a sequence of participating videos. However, it additionally discloses the sponsoring firm and shows a list official videos on top created by its sponsored influencers, on top of all other free videos created by regular users.

G.1. Differences on Observed Characteristics

First, we verify that sponsored hashtags are similar to non-sponsored hashtags in terms of their observable characteristics, including which types of influencers tend to post with them, and the type of content posted. In Table A-7, we show the difference in means between sponsored and organic hashtags for all of our control variables described in Web Appendix A. These variables include the post metadata, creator metadata, and the creator features we extracted directly from the video (but did not use in the main analysis; see Web Appendix A.3). We can see, across all of these variables, that sponsored and organic hashtags are quite similar. None of the differences are significant at the 0.05 level.[7] This analysis suggests that sponsored and organic hashtags are comparable on all observed dimensions.

[7]The only difference that stands out in terms of magnitude is video age: this difference is driven by a few organic hashtags like #NBADraft or #VeteransDay, which occur every year. For these annual hashtags, there are a tiny fraction of videos that appear on the Discover page from previous years. This small number of old videos drives up the average age.

Variable	Sponsored Mean	Organic Mean	Mean Difference
Metadata			
Num Hashtags	5.598	6.774	-1.176
Num Trending	0.9045	1.135	-0.231
Has FYP	0.5114	0.652	-0.141
Hashtag Age	3.466	6.1930	-2.727
Is Trending	0.849	0.758	0.091
Ranking	934.4	1048	-113.6
Video Age	2.0295	35.446	-33.417
Is sponsored	0.006	0.0009	0.0051
log(Following)	2.012	2.184	-0.172
log(Num_Videos)	2.145	2.099	0.055
Is verified	0.007	0.002	0.005
Gender			
Male	0.349	0.396	-0.047
Female	0.651	0.604	0.047
Age Group			
10-19	0.352	0.381	-0.029
20-29	0.495	0.473	0.022
30-39	0.140	0.115	0.025
40-49	0.009	0.021	-0.012
50+	0.004	0.010	-0.006
Race			
Asian	0.030	0.072	-0.042
Black	0.287	0.254	0.033
Indian	0.012	0.063	-0.051
Latino	0.357	0.342	0.015
Middle Eastern	0.009	0.029	-0.02
White	0.305	0.240	0.065
Emotion			
Angry	0.138	0.126	0.012
Fear	0.004	0.011	-0.007
Neutral	0.010	0.012	-0.002
Sad	0.125	0.166	-0.041
Disgust	0.193	0.224	-0.031
Happy	0.369	0.341	0.028
Surprise	0.161	0.120	0.041

Table A-7: Mean Difference for Control Variables between Sponsored and Organic Hashtags
The difference in means between our control variables for sponsored versus organic hashtags. Here, we include the creator variables we extracted directly from the videos, though we again note that those variables are not used in our model. None of the differences are significant at the 0.05 level.

G.2. Influencer Self-selection

To further establish the comparability between organic and sponsored hashtags, we examine whether influencers with different follower sizes might be self-selecting into creating content under sponsored vs. organic hashtags. In Figure A-4, we plot the distributions of influencers by influencer tiers between sponsored vs. organic hashtags. We find that the overall distributions are consistent: mid-tier influencers take the largest share, followed by micro- and nano-influencers. However, we do note that a slightly higher percentage of mega-influencers choose to create videos under sponsored hashtags, while a slightly higher percentage of the smaller influencers (i.e., sub-nano, nano, and micro) post under organic hashtags. This discrepancy is likely due to the common firm strategy of sponsoring very popular influencers.

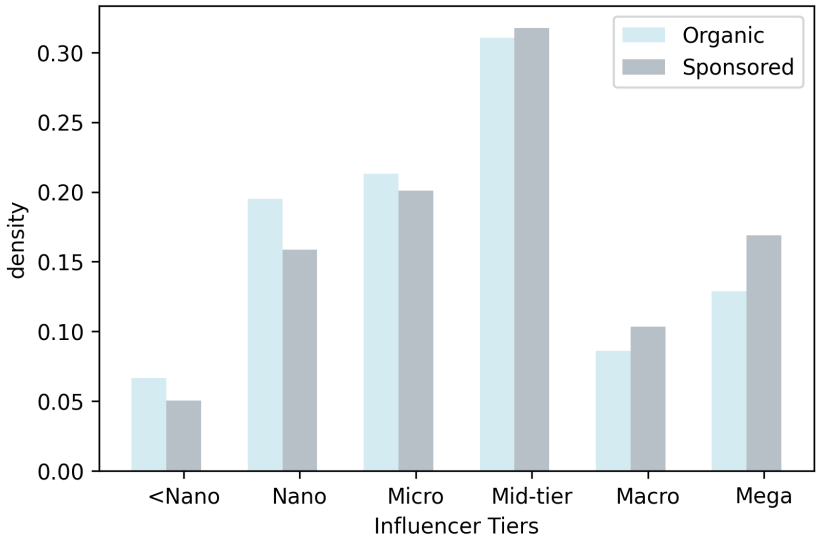


Figure A-4: Distribution of Influencer Popularity Between Sponsored and Organic Hashtags

H. Classifying Sponsored Campaigns Curve Types

To classify each of the sponsored campaigns into curve types, after learning the FEI curve for each video, we compute the average FEI over six ranges, corresponding to the six tiers of influencers: sub-nano (fewer than 1,000 followers), nano (1-10K followers), micro (10-50K), mid-tier (50-500K), macro (0.5-1M), or mega (more than 1 million followers). Table A-8 reports these average FEIs for each campaign.

	<Nano	Nano	Micro	Mid-tier	Macro	Mega
S-shaped						
#UnwrapTheDeals	0.08	0.37	0.61	0.33	0.08	0.02
#WhatsYourPower	0.12	0.53	0.48	0.22	0.13	0.06
#VideoSnapChallenge	0.16	0.52	0.63	0.25	0.11	0.03
#GetCrocd	0.09	0.44	0.59	0.35	0.10	0.02
Linear						
#CaliStarChallenge	0.20	0.23	0.25	0.28	0.22	0.20
#HandWashChallenge	0.27	0.24	0.22	0.24	0.18	0.20
#MoreHappyDenimDance	0.15	0.18	0.22	0.19	0.20	0.17
#ScoobDance	0.10	0.14	0.18	0.16	0.13	0.11
#ASOSFashunWeek	0.30	0.35	0.31	0.28	0.31	0.26
#TheSplashDance	0.22	0.24	0.28	0.25	0.23	0.19
#ReadySETgo	0.25	0.30	0.27	0.29	0.24	0.21
#LetsFaceIt	0.27	0.33	0.35	0.32	0.30	0.28
#DoPacSun	0.29	0.32	0.28	0.26	0.23	0.22
#MerryBOSSmas	0.17	0.18	0.24	0.16	0.15	0.14
#MoodFlip	0.19	0.26	0.23	0.22	0.20	0.19
#ThisIsBliss	0.28	0.36	0.31	0.33	0.32	0.30
#UptheBeat	0.22	0.24	0.27	0.23	0.22	0.21
#katespadenyhappydance	0.16	0.19	0.17	0.20	0.21	0.18
#PerfectAsIAm	0.24	0.32	0.25	0.28	0.30	0.27
#exprESSIEyourself	0.16	0.22	0.20	0.20	0.23	0.19
Concave						
#CloseYourRings	0.36	0.17	0.15	0.11	0.08	0.05
#ItWasntMe	0.39	0.30	0.22	0.15	0.08	0.05
#HeinzHalloween	0.47	0.33	0.19	0.12	0.06	0.04
#StrictlyCurl	0.63	0.37	0.24	0.18	0.11	0.06
#MONCLERBUBBLEUP	0.41	0.26	0.15	0.11	0.06	0.02
#ShowUpShowOff	0.58	0.43	0.27	0.13	0.10	0.02
#GotMilkChallenge	0.47	0.36	0.23	0.14	0.09	0.03
#GoForTheHandful	0.64	0.42	0.28	0.13	0.11	0.06
#MicellarRewind	0.39	0.30	0.21	0.12	0.10	0.04
#CancelTheNoise	0.42	0.35	0.22	0.15	0.12	0.05

Table A-8: Campaign FEI by Influencer Tiers

For each sponsored challenge, we compute the average of the FEI curve of that challenge over six intervals, corresponding to different influencer tiers. We then grouped these curves into three buckets, based on the implied shape of the corresponding counterfactual prediction curve: S-shaped, linear, and concave.

I. Video Content and Popularity

On TikTok's Discover page, influencers are presented with trending hashtags that encourage them to create videos based on specific themes. However, video-making skills and experience may vary across influencers. This variation raises a potential concern of self-selection, wherein very experienced influencers may choose to post under certain hashtags, while inexperienced or less professional (and therefore, potentially less popular) influencers may post under different, less demanding hashtags. Thus, in this appendix, we investigate whether any such correlation exists between an influencer's video content and their follower count across different hashtags.

To accomplish this, we follow the same procedure as in other parts of the paper and categorize influencers into sub-nano, nano, micro, mid-tier, macro, and mega based on their follower counts. We then compare their VAE representations across these tiers. Specifically, since our VAE representations are high-dimensional, we employ t-SNE (Van der Maaten and Hinton 2008), a nonlinear dimensionality reduction technique, to map the content representation of all videos under a hashtag into a two-dimensional space. To see if these representations vary by popularity, we plot these two-dimensional embeddings and label each point according to its creator's popularity tier. We repeat this process for all campaign hashtags. We display four randomly selected examples of these plots in Figure A-5. Our results show that there are no significant correlations between influencer tiers and video content clusters. However, the distribution of video content under different campaign hashtags does exhibit distinct patterns. Some hashtags display clear clusters, which may result from influencer segmentation in other dimensions such as early vs. late participants or different trends within the hashtag, rather than follower count.

To further understand whether there is any link between influencer popularity and the type of hashtags they tend to contribute to, we return to the three types of sponsored campaigns that we described in the main body of the paper: special effects, self-expression, and product demonstration. In Figure A-6, we plot the distribution of influencer tiers across the three campaign types. We find distributions of types roughly consistent across campaign type, with mid-tier influencers having the largest share, followed by micro- and nano-influencers. All types of influencers are represented in all campaigns. The only notable imbalance is the proportion of mega influencers in self-expression campaigns, which is modestly higher.

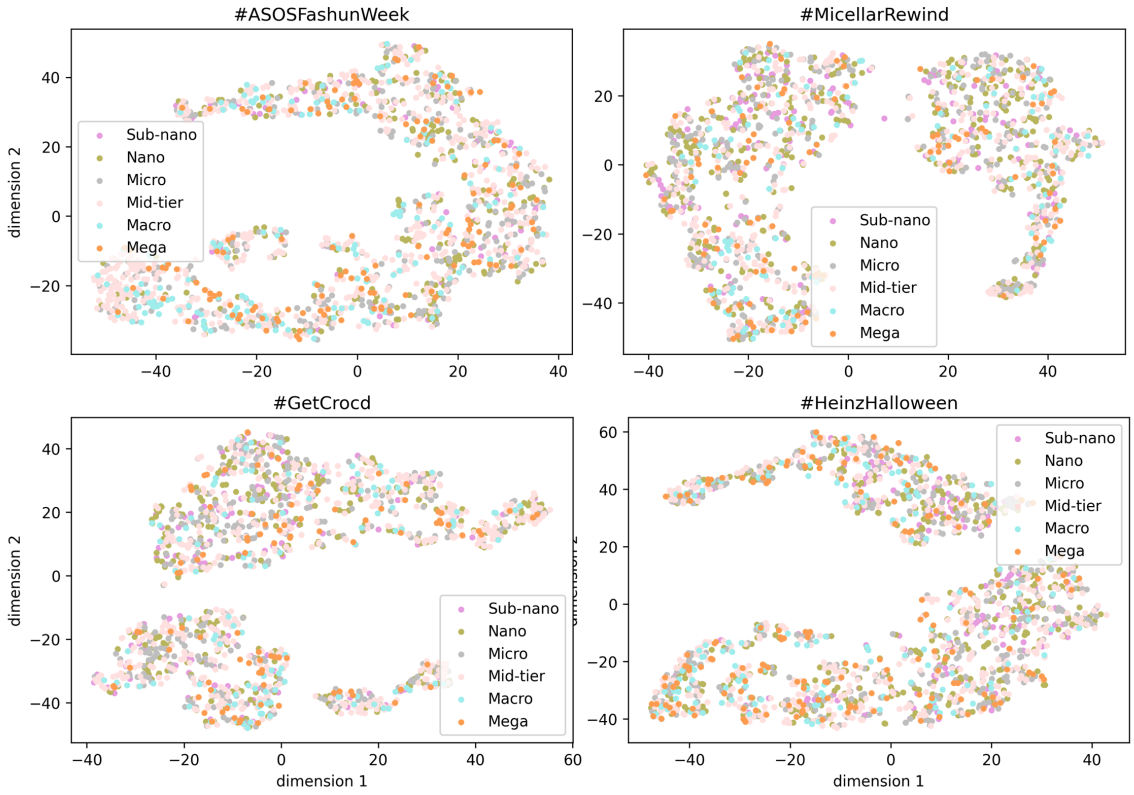


Figure A-5: 2D Visualization of Video Representations under a Random Sample of Hashtags

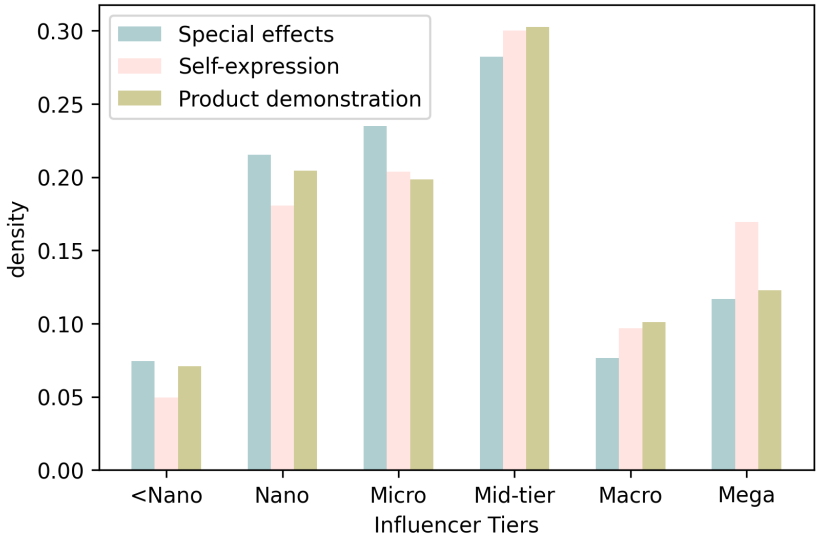


Figure A-6: Distribution of Influencer Tiers by the Three Sponsored Campaign Types

Taken together, our findings suggest that influencers across different tiers do not tend to self-select into specific types of content or hashtags. In some sense, this is not surprising: many trending hashtags, especially for sponsored campaigns, require little expertise for people to create

content under them, in order to encourage a larger community to participate. For example, all influencers in the Walmart campaign were able to create similar videos following the hashtag-suggested format (e.g., facing the camera, grabbing the gift box on top of their heads, and unwrapping the deals inside).

J. Walmart Profit Maximization

We provide additional details about the Walmart case study. Our goal is to determine the optimal popularity level for an influencer for Walmart's #UnwrapTheDeals campaign. As noted in the main text, doing so requires three ingredients: 1) the predicted FEI curve, from which we get the marginal returns to a follower in terms of impression for that campaign; 2) an assumption about how impressions translate into revenue; and 3) the cost structure of the sponsorship.

To obtain the predicted FEI curve for #UnwrapTheDeals, we first estimate the FEIs for all the #UnwrapTheDeals sponsored videos. We then average their FEIs to get the predicted FEI curve for the overall campaign.[8] The FEI tells us the percentage gain in impressions given a 1% gain in followers. From this, we can also straightforwardly compute the marginal effect of adding a single follower, rather than a percentage.

To understand how these marginal effects translate to revenue, we need to make an assumption about how impressions relate to revenue. For our case study, we assume a constant (linear) return of \$0.02 per impression, which is based on the average cost per view of a video ad on YouTube.[9] Multiplying this dollar per impression rate by the number of impressions per follower calculated in step 1 yields the marginal revenue for investing in an influencer of a given popularity. We plot this curve for #UnwrapTheDeals in Figure A-7. We note that the marginal revenue curve is essentially the mirror image of the counterfactual prediction function in the main text, and predictably exhibits diminishing returns.

To understand the optimal popularity, we need to make assumptions about costs, from which we can evaluate the profit-maximizing first-order conditions (i.e., marginal cost = marginal revenue). There are many payment schemes firms used when contracting influencers, but they are typically based on the number of followers the influencer has.[10] One simple scheme is to pay influencers linearly, usually in increments of followers (e.g., per 1,000 followers). Industry reports suggest that these rates can go as high as \$100 per 1,000 followers,[11] but a more typical rate on

[8] Note that there is very little variation in the FEI curves across these videos: they were all created following the same template suggested by the campaign (e.g., facing in front of the camera, grabbing the gift box from Walmart on top of their heads, and unwrapping the deals inside), and their content is very similar. As a result, their representations and, thus, FEI curves are also very similar.

[9] https://influencermarketinghub.com/how-much-do-youtube-ads-cost/

[10] E.g., https://influencermarketinghub.com/influencer-rates/

[11] https://influencermarketinghub.com/influencer-marketing-statistics/

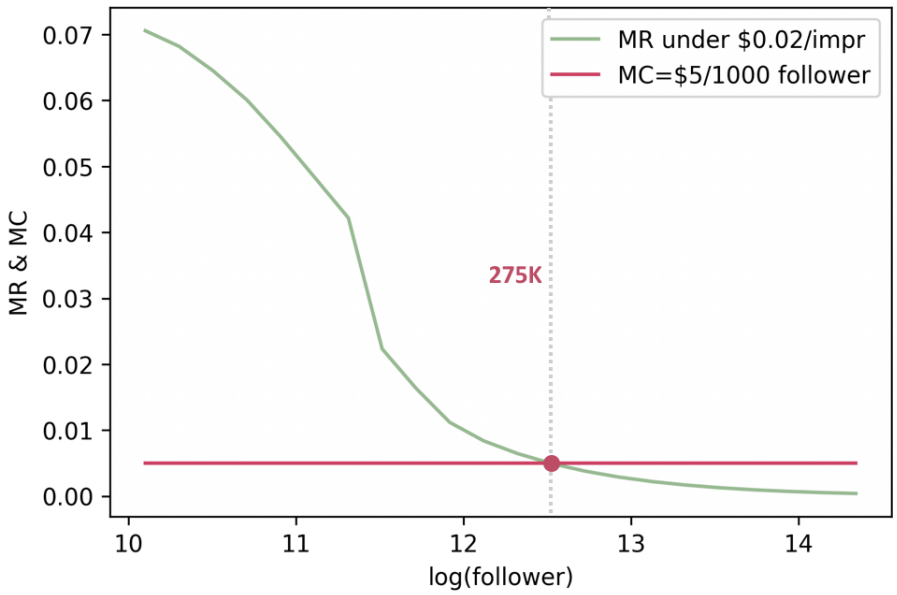


Figure A-7: Marginal revenue and marginal cost of with respect to (log) followers

TikTok is around $5 per 1,000 followers.[12] This is what we assume for the firm's marginal cost, and plot as the horizontal line in Figure A-7.

Having obtained both the marginal revenue of sponsoring an influencer to create a video for this Walmart campaign and its marginal cost, we can now determine the optimal popularity by simply evaluating the profit maximization first-order conditions. In Figure A-7, this is simply the intersection point of marginal cost and marginal revenue. Given our assumptions, we find that the optimal follower size for #UnwrapTheDeals is 275,000. The profit in this scenario is 56% more than paying a typical micro influencer with 50,000 followers, and 300% more than paying a mega influencer with 1.5M followers (which was Walmart's choice in reality), as illustrated in Figure A-8.

[12] https://www.refersion.com/blog/influencer-marketing-cost/

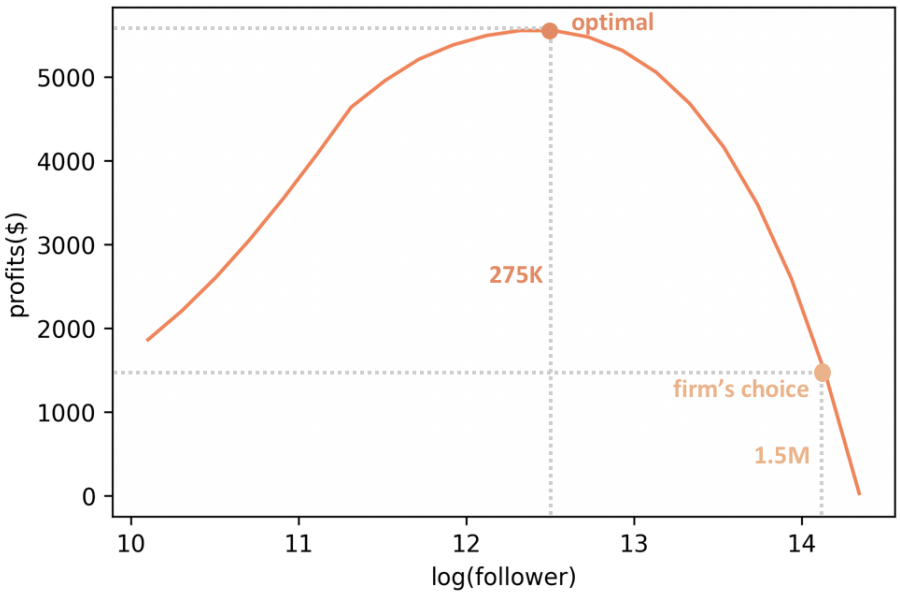


Figure A-8: Walmart's #UnwrapTheDeals Profits by Popularity

References

Bhattacharya, Subhabrata, Behnaz Nojavanasghari, Tao Chen, Dong Liu, Shih-Fu Chang, and Mubarak Shah "Towards a Comprehensive Computational Model Foraesthetic Assessment of Videos," "Proceedings of the 21st ACM International Conference on Multimedia," pages 361–364 (2013).

Borth, Damian, Rongrong Ji, Tao Chen, Thomas Breuel, and Shih-Fu Chang "Large-scale Visual Sentiment Ontology and Detectors Using Adjective Noun Pairs," "Proceedings of the 21st ACM International Conference on Multimedia," pages 223–232 (2013).

Dew, Ryan, Asim Ansari, and Olivier Toubia (2022), "Letting Logos Speak: Leveraging Multiview Representation Learning for Data-driven Branding and Logo Design," *Marketing Science*, 41 (2), 401–425.

Gelman, Andrew, Ben Goodrich, Jonah Gabry, and Aki Vehtari (2019), "R-squared for Bayesian Regression Models," *The American Statistician*.

Gemmeke, Jort F., Daniel P.W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter "Audio Set: An Ontology and Human-labeled Dataset for Audio Events," "2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)," pages 776–780, IEEE (2017).

Hartford, Jason, Greg Lewis, Kevin Leyton-Brown, and Matt Taddy (2017), "Deep IV: A Flexible Approach for Counterfactual Prediction," *Proceedings of the International Conference on Machine Learning*, pages 1414–1423.

Hershey, Shawn, Sourish Chaudhuri, Daniel P.W. Ellis, Jort F. Gemmeke, Aren Jansen, R. Channing Moore, Manoj Plakal, Devin Platt, Rif A. Saurous, Bryan Seybold et al. "CNN Architectures for Large-scale Audio Classification," "2017 IEEE International Conference on Acoustics, Speech and Signal Processing (icassp)," pages 131–135, IEEE (2017).

Kingma, Diederik P and Jimmy Ba (2014), "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean "Distributed Representations of Words and Phrases and Their Compositionality," "Advances in neural information processing systems," pages 3111–3119 (2013).

Newey, Whitney K. and James L. Powell (2003), "Instrumental Variable Estimation of Nonparametric Models," *Econometrica*, 71 (5), 1565–1578.

Schuster, Mike and Kuldip K. Paliwal (1997), "Bidirectional Recurrent Neural Networks," *IEEE transactions on Signal Processing*, 45 (11), 2673–2681.

Simonyan, Karen and Andrew Zisserman (2014), "Very Deep Convolutional Networks for Large-scale Image Recognition," *arXiv preprint arXiv:1409.1556*.

Taigman, Yaniv, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf "Deepface: Closing the Gap to Human-level Performance in Face Verification," "Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition," pages 1701–1708 (2014).

Van der Maaten, Laurens and Geoffrey Hinton (2008), "Visualizing Data Using t-SNE," *Journal of Machine Learning Research*, 9 (11).

Wei, Yinwei, Xiang Wang, Weili Guan, Liqiang Nie, Zhouchen Lin, and Baoquan Chen (2019), "Neural Multimodal Cooperative Learning toward Micro-video Understanding," *IEEE Transactions on Image Processing*, 29, 1–14.

Zha, Shengxin, Florian Luisier, Walter Andrews, Nitish Srivastava, and Ruslan Salakhutdinov (2015), "Exploiting Image-trained CNN Architectures for Unconstrained Video Classification," *arXiv preprint arXiv:1503.04144*.