Tamper-proofing Video with Hierarchical Attention Autoencoder Hashing on Blockchain

Tu Bui, Daniel Cooper, John Collomosse, Mark Bell, Alex Green, John Sheridan, Jez Higgins, Arindra Das, Jared Robert Keller and Olivier Thereaux

***Abstract*—We present ARCHANGEL; a novel distributed ledger based system for assuring the long-term integrity of digital video archives. First, we introduce a novel deep network architecture using a hierarchical attention autoencoder (HAAE) to compute temporal content hashes (TCHs) from minutes or hour-long audio-visual streams. Our TCHs are sensitive to accidental or malicious content modification (tampering). The focus of our self-supervised HAAE is to guard against content modification such as frame truncation or corruption but ensure invariance against format shift (*i.e.* codec change). This is necessary due to the curatorial requirement for archives to format shift video over time to ensure future accessibility. Second, we describe how the TCHs (and the models used to derive them) are secured via a proof-of-authority blockchain distributed across multiple independent archives. We report on the efficacy of ARCHANGEL within the context of a trial deployment in which the national government archives of the United Kingdom, United States of America, Estonia, Australia and Norway participated.**

***Index Terms*—Distributed Ledger Technology, Content aware hashing, autoencoder, LSTM, attention network, content integrity, blockchain.**

I. Introduction

ARCHIVES are the lens through which future generations will perceive the events of today. Increasingly those events are captured in digital video form, raising new challenges around the assurance of trust, immutability and long term accessibility of digital video records [1]. Digital video formats are ephemeral requiring regular format shifting (video transcoding) as part of an archive's curatorial duty to keep content accessible as the years go by. Additionally, the volume and intangibility of digital video leaves it open to modification (tampering) – either due to malicious attack, or accidental corruption during bulk transcoding processes.

This paper proposes ARCHANGEL; a novel de-centralised system to guard against tampering of digital video within archives, using a permissioned blockchain maintained collaboratively by several independent archive institutions. We propose a novel deep neural network (DNN) architecture trained to distil an audio-visual signature (content hash) from video, that is sensitive to tampering, but invariant to the format *i.e.* the video codec used to compress the video. Video signatures are computed and stored immutably within the ARCHANGEL blockchain at the time of the video's ingestion to the archive. The video can be verified against its original signature at any time during curation, including at the point of public release, to ensure the integrity of content.

The motivation of this paper is based on the needs for national government archives to retain video records for years or even decades. For example, records of the supreme court proceedings in the United Kingdom are born-digital. These records are held in the National Archives for 5 years prior to release, upon which they form precedent within the legal system. It is crucial that such content, archived today, is the same as that released in future; and regardless of length or format no modification of the audio or visual streams should be possible. Yet since the first digital video standard, H.120, was introduced in 1984, hundreds of coding standards (each supported by various file formats) have been developed [2]. The original archival video, therefore, will unlikely be the same as the one presented to the user. This renders bit-level cryptographic hashes like SHA-256 [3] impractical in securing the authenticity of video archives. This motivates *content-aware* hashing of the audio-visual stream within the video. We therefore propose two technical contributions:

1) Temporal Content Hashing. A novel DNN based temporal content hash (TCH) that is trained to ignore transcoding artifacts, but is capable of detecting tampers of a few seconds duration within typical video clip lengths in the order of minutes or hours. The hash is computed over both the audio and the visual components of the video stream. We initially proposed an earlier version of our TCH scheme using a hybrid CNN-LSTM network in workshop paper [4]. In this extended paper we build on this approach to propose HAAE – a hierarchical LSTM architecture that also incorporates a novel attention scheme. This model yields superior performance and removing the need to produce several models for a single stream. We also explore applications of our hierarchical attention model to alternative recurrent networks beyond LSTM and evaluate alternative CNN backbones.

2) Cross-archive Blockchain for Video Integrity We propose the storage of TCHs within a permissioned proof-of-authority blockchain maintained across multiple independent archives participating in ARCHANGEL, enabling mutual assurance of video integrity within their archives. Fine-grain temporal sensitivity of the TCH is challenging given the requirement of

Manuscript received August 20, 2019; revised December 20, 2019; accepted January 13, 2020. This work is supported by Archangel, the 2-year EPSRC Research Grant EP/P03151X/1.

T. Bui and D. Cooper are with the Centre for Vision Speech and Signal Processing (CVSSP), University of Surrey, Guildford GU2 7XH, UK (e-mail: {t.bui,d.cooper}@surrey.ac.uk).

J. Collomosse is with the University of Surrey, Guildford GU2 7XH, UK and Adobe Research, San Jose, CA 95110, USA (e-mail: j.collomosse@surrey.ac.uk).

M. Bell, A. Green and J. Sheridan are with The National Archives, Kew, Richmond TW9 4DU, UK.

J. Higgins, A. Das, J.R. Keller and O. Thereaux are with The Open Data Institute, Clifton Street, London EC2A 4JE, UK.

hash compactness for scalable storage on the blockchain. We therefore take a hybrid approach. The codec-invariant TCHs computed by the model are stored compactly on-chain. The model itself is stored off-chain alongside the source video (*i.e*. within the archive), and hashed on-chain via SHA-256 to mitigate attack on the TCH computation.

We demonstrate the value of ARCHANGEL through a trial international deployment across the national government archives of the United Kingdom, United States of America, Australia, Norway and Estonia. Collaboration between the majority of partnering archives ('50% attack' [5]) would be needed to rewrite the blockchain and so undermine its integrity, which is unlikely between archives of independent sovereign nations. This justifies our use of Blockchain, which has not been previously combined with temporal content-hashing to ensure video archive integrity.

II. Related Work

Content-aware video hashing is a long-studied problem. Classical approaches to visual hashing explored heuristics to extract video feature representations including discrete cosine transform [6], spectral descriptors [7], and robust temporal matching of visual structure [8], [9], [10] for video near-duplicate detection/retrieval. Sivic *et al*. [11], [12] used a shallow learning approach based on sparse gradient-domain features for object retrieval within videos.

The advent of deep learning delivered highly powerful hashing algorithms. Convolutional neural networks (CNN) were largely used to encode spatial content [13], [14], [15], [16]. Gou *et al*. [13] used ResNet [17] to encode video and even audio features. Xu *et al*. [14] proposed to aggregate CNN frame features with VLAD encoding for event detection, sacrificing temporal coherence as the whole. Liong *et al*. [15] attempted to encode motion-based features via average pooling of consecutive frame features, effectively modelling temporal sequence as short as 10 frames while semantic features were learned in a supervised manner. Alternatively, Kim *et al*. [16] employed 3D CNN to model space-time features in an end-to-end network for video recognition; however due to high memory requirements their proposed network could only model low resolution video sequences of up to 80x80x16.

To model long sequences more advanced networks that exhibit temporal dynamic behaviour such as recurrent neural networks (RNN) are often employed [18]. Time Delay Neural Networks (TDNN) based on Tapped Delay Line (TDL) were applied in phoneme recognition and energy disaggregation [19], [20], [21]. Gated Recurrent Unit (GRU) was employed in machine translation [22] while Long Short-term Memory networks (LSTM) have been used extensively for video retrieval in either supervised [23] or unsupervised manner with binarization constraints [24]. Unsupervised LSTM are also achievable with deep auto-encoders [25], [26], [27]. Other works focus on renovating LSTM architectures, resulting in stacked LSTM [24], hierarchical LSTM [27], bidirectional LSTM [28] and multi-head LSTM [25]. These techniques demonstrate impressive performance on specific tasks and motivate the design of our HAAE.

Video tampering detection is an important topic for digital archives but unfortunately largely unexplored in literature. The emergence of high-realism video manipulation using deep learning ('deep fakes' [29]) has renewed interest in determining the provenance of digital video. Existing approaches often focus on specific types of tampering *e.g*. frame insertion/deletion/duplication [30] or copy-paste forgery [31], [32] by analyzing low level patterns such as Fourier transform signals or noise distribution. These techniques often work with strict assumptions of camera settings, lighting conditions, motion/background dynamics, and even video length and codecs [33]. Very few deep learning approaches were proposed for tampering detection, using either 3D CNN [34] or RNN [35], [36], [37] but having the same limitations regarding video length and task-specific. Other different but related topics include anomaly/outlier detection [38], [39] and novelty detection/one-class classification [40] that involves learning space-time visual cues within videos tailored for the corresponding tasks.

All above approaches mostly focus on the task of detecting visual manipulation or abnormality, rather than immutable storage of video hashes over longitudinal time periods, as discussed in this paper. Our work does not tackle the task of detecting video manipulation *ab initio* (we trust video at the point of ingestion). Rather, our approach can detect subsequent tampering with a video, offering a promising tool to verify video integrity via proof of content provenance. Moreover, existing techniques train a global model for content hashing over a representative video corpus, and focus on hashing short clips with lengths of a few minutes at most using visual cues only. Audio-visual video fingerprinting via multi-modal analysis is sparsely researched with prior work focusing upon augmenting visual matching with an independent audio fingerprinting step [41], predominantly via wavelet analysis [42], [43]. Our archival context requires hashing of diverse video content; training a single representative DNN to hash such videos is unlikely practical nor future-proof. Rather our work takes an unsupervised approach, fitting a HAAE network model to a single video to minimise frame reconstruction and prediction error. This has the advantage of being able to detect short duration tampering within long duration video sequences (but at the overhead of storing a model per hashed video). Since videos in our archival context may contain limited visual variation (*e.g*. a committee or court hearing) we hash both video and audio modalities. Uniquely, we expose the model to variations in video compression during training to avoid detecting transcoding artifacts as false positives.

Distributed Ledger Technology (DLT) has existed for over a decade - notably as blockchain, the technology that underpins Bitcoin [44]. The unique capability of DLT is to guarantee the integrity and provenance of data when distributed across many parties, without requiring those parties to trust one another nor a central authority [5]; *e.g*. Bitcoin securely tracks currency ownership without relying upon a bank. Yet, the guarantees offered by DLT for distributed, tamper-proof data have potential beyond the financial domain [45], [46]. We build upon recent work suggesting the potential of DLT for digital record-keeping [47], [48], notably an earlier

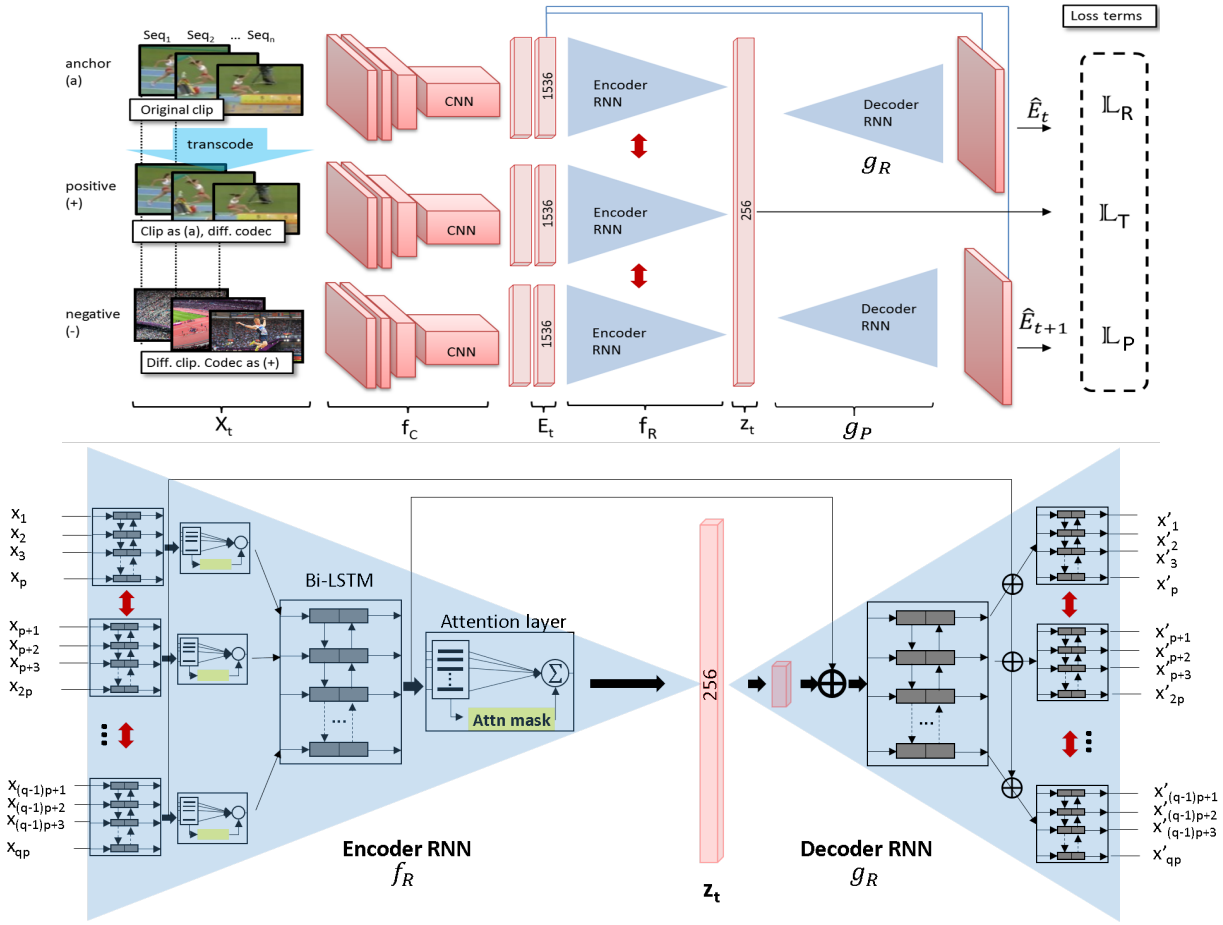


Fig. 1. Our HAAE network for video tamper detection. (Top) this schematic is for visual cues, using InceptionResNetV2 backbone (f_C) to encode video frames (audio network uses MFCC features instead). Once extracted, features are passed through the RNN encoder-decoder network where codec-invariant features are learnt via a set of loss functions. (Bottom) Hierarchical layout of the encoder (f_R) and decoder RNNs. Each layer of f_R comprises a bidirectional LSTM module followed by an attention module. The attention module is omitted in the decoder part. Bottleneck z_t serves as the content hash and is compressed via PQ post-process (not shown).

incarnation of ARCHANGEL described in Collomosse *et al.* [47], [4] which utilised a proof-of-work (PoW) blockchain to store SHA-256 hashes of binary zip files containing academic research data. Also related, is the prior work of Gipp *et al.* [49] where SHA-256 hashes specifically of video are stored in a PoW (Bitcoin) blockchain for evidencing car collision incidents. Our framework differs, due to the insufficiency of such binary hashes [3] to verify the immutability of video content in the presence of format shifting, which in turn demands novel content-aware video hashing, and a hybrid on- and off-chain storage strategy for safeguarding those hashes (and DNN models generating them).

III. Methodology

We propose a content hashing framework for integrity assurance of archival videos within several National Archives. Our framework accepts a digital video of arbitrary length as input and outputs at max two sets of compactly codec-invariant hashes, one for the video stream and the another for audio (if the audio stream is also available). The hashes (which we from now refer as *temporal content hashes*, TCH) encode spatio-temporal video features and can be used to detect even minor spatial (frames) corruption or temporal discontinuities (truncation, splices).

The core of our TCH framework is a novel self-supervised recurrent neural networks (RNN) which is trained such that reformatted (transcoded) videos produce near-identical TCHs after a feed forward pass through the models, yet a tampered video results in a different TCH. Note that we do not train a single model to encode all videos, but instead fit a video-specific model to content. This approach is advantageous because (i) a well-performing model can be trained for an extremely-long or highly static video and (ii) zero assumptions on what constitutes the representative content of an archive *i.e.* each video is the training data of its own model. A disadvantage of this approach is that the model must be stored alongside the video as meta-data in the archive. This measure is necessary to prevent attacks on the TCH verification process.

We describe the architecture of our TCH network in sub-sec. III-A. Details of the hash extraction, compression and tampering detection processes are discussed in subsec. III-B.

A. Hierarchical Attention Auto-Encoder (HAAE) Network

We propose a novel triplet convolutional-recurrent network structure resembling an autoencoder (AE). Frame-level spatial

features are extracted using a CNN network before being fed into an AE network to encode its temporal coherence. Fig. 1 (top) illustrates the overall architecture for encoding visual content (audio content is discussed in subsec. III-A5).

1) Spatial encoding: Given a continuous video clip $\mathcal{X}$ of arbitrary length we sample keyframes at 10 frames per second (fps). Keyframes are aggregated into $T = \frac{|\mathcal{X}|}{N}$ blocks where $N = 150 - 600$ frames depending on $|\mathcal{X}|$ ($\sim$ 15-60 seconds, padded with empty frames if necessary in the final block; see subsec. IV-B for more details), yielding a series of sub-sequences $\mathcal{X} = \{X_1, X_2, ..., X_T\} \in \mathbb{R}^{N\times H\times W\times C}$ where H, W and C are the height, width and number of channels for each image frame (usually C=1 or 3 for gray or color RGB frame). A CNN network encodes each block X_t to a sequence of semantic features that summarize spatial content. $f_C : \mathbb{R}^{H\times W\times C} \rightarrow \mathbb{R}^S$, which encodes $\mathcal{X}$ into $\mathcal{E} = \{E_t \in \mathbb{R}^{N\times S} \mid t = 1, 2, ..., T\}$ with S being the output dimension of the final fully connected (FC) layer. Our proposed model uses off-the-shelf InceptionResNetV2 backbone [50] for f_C ($S = 1536$). An ablation study about CNN architectures is discussed in subsec. IV-C.

2) HAAE: We propose to model the temporal relationship between consecutive frame features E_t with an RNN, $f_R : \mathbb{R}^{N\times S} \rightarrow \mathbb{R}^D$, resulting in an embedding $\mathcal{Z} = \{z_t \in \mathbb{R}^D \mid t = 1, 2, ..., T\}$. Fig. 1 (bottom) illustrates the architecture of f_R which consists of 2 layers each having a Bi-directional Long Short Term Memory (Bi-LSTM) module followed by an attention module.

Denote e_{it} the ith feature vector, $i \in [1, N]$, of the tth block E_t, $t \in [1, T]$. A *standard* LSTM module takes in input vector e_{it} and previous state $h_{i-1,t}$ to output current state h_{it}:

$$[j_i, g_i, u_i, o_i]^T = \boldsymbol{W} e_{it} + \boldsymbol{U} h_{i-1,t} + \boldsymbol{b}, \quad (1)$$

$$c_i = c_{i-1} \odot \sigma(g_i) + tanh(u_i) \odot \sigma(j_i), \quad (2)$$

$$h_{it} \coloneqq LSTM(e_{it}) = \sigma(o_i) \odot tanh(c_i). \quad (3)$$

where j_i, g_i, u_i, o_i, c_i are input gate, forget gate, candidate activation, output gate and cell state at time step i respectively; $\sigma(.)$ is the sigmoid function and $\odot$ indicates the element-wise product; $\boldsymbol{W} = \{W_j, W_g, W_u, W_o\}, \boldsymbol{U} = \{U_j, U_g, U_u, U_o\}, \boldsymbol{b} = \{b_j, b_g, b_u, b_o\}$ are sets of learnable weight matrices and bias parameters for the corresponding gates.

A ***Bi-LSTM*** module parses an input sequence $E_t = \{e_{it} \mid i = 1, 2, ...N\}$ in both forward and backward directions, concatenating hidden states as outputs:

$$\overrightarrow{h}_{it} = \overrightarrow{LSTM}(e_{it}), \ i \in [1, N] \quad (4)$$

$$\overleftarrow{h}_{it} = \overleftarrow{LSTM}(e_{it}), \ i \in [N, 1] \quad (5)$$

$$h_{it} = [\overrightarrow{h}_{it}, \overleftarrow{h}_{it}]. \quad (6)$$

This results in a 2x-dimensional output as compared with standard LSTM. However Bi-LSTM often outperforms standard LSTM as it explores both past and future context concurrently. In our experiments, we observe that enabling Bi-LSTM makes the training more stable, especially on static videos where changes in subsequent time steps are insignificant.

The Bi-LSTM outputs are weight-summed to a single representation vector, z_t, via an attention module. Specifically,

$$v_{it} = tanh(W_a h_{it} + b_a), \quad (7)$$

$$\alpha_{it} = \frac{e^{v_{it}^T v_a}}{\sum_i e^{v_{it}^T v_a}}, \quad (8)$$

$$z_t = \sum_i \alpha_{it} h_{it}. \quad (9)$$

That is, the input of the attention module, h_{it}, is first projected into a hidden embedding space via a FC layer, becoming v_{it}. The importance of v_{it} is measured via its similarity with a context vector, v_a, which is also learnable. We then normalise this importance factor through a softmax function, resulting in $\alpha_{it} \in \mathbb{R}$. Finally, inputs $h_{it} \mid i = 1, ..., N$ are accumulated with weights α_{it} to produce output z_t which is the bottleneck (embedding) of our HAAE network.

Hierarchical LSTM: although there is no obvious hierarchical structure in video frames (unlike text having a clear hierarchical order of characters, words, sentences and so on), we find a hierarchical design of f_R advantageous in modelling long sequence. That is, assume $N = p \times q \mid p, q \in \mathbb{N}^+$, we split block E_t into q sub-blocks, $E_t = \{E_{it} \in \mathbb{R}^{p\times S} \mid i = 1, 2, ..., q\}$, then pass these sub-blocks into the first Bi-LSTM/Attention layer. This results in q output vectors which form the input sequence for the second Bi-LSTM/Attention layer (see also Fig. 1, bottom):

$$r_{it} = f_{R1}(E_{it}), \ i \in [1, q], \quad (10)$$

$$R_t = \{r_{it} \mid i = 1, 2, ..., q\} \in \mathbb{R}^{q\times S'}, \quad (11)$$

$$z_t = f_{R2}(R_t) \in \mathbb{R}^D. \quad (12)$$

where f_{R1}, f_{R2} denotes the two Bi-LSTM/Attention layers and S' is the intermediate output dimension. Each layer of f_R now processes much shorter sequences thus can learn a finer representation. We study the performance of our approach versus one-layer RNN in subsec. IV-C.

Note that our design of f_R is motivated from the work of Yang *et al*. [51] however their network is for text classification whereas f_R is an encoder part of our larger HAAE network. Additionally, our design differs from stacked LSTM [24] in term of modularity – in our approach each sub-block E_{it} is processed independently while in stacked LSTM the notation of sub-block does not exist. Also our approach results in a much smaller model as compared with stacked LSTM.

Decoder: our HAAE network has two decoders, g_R and g_P, hierarchically mirror the encoder design. g_R is designed to reconstruct the input sequence E_t while g_P predicts the next sequence E_{t+1}. We omit the attention module in g_R and g_P (since a decoder "expands" features while an attention layer migrates them). For g_R, we bridge the outputs of each layer in the encoder (just before the attention module) to the corresponding layer of the decoder, effectively resembling a U-Net structure [52]. We use 'average' bridge instead of concatenation as in [52] and find this equally effective but resulting in a smaller network. For g_P, bridging is not necessary and so does the bidirectional LSTM. Therefore we use hierarchical standard LSTM to model this branch.

Specifically, for the reconstruction decoder g_R:

$$z'_t = W' z_t + b', \tag{13}$$

$$r^{R2}_{it} = \frac{1}{2}(z'_t + h^{E2}_{it}), \tag{14}$$

$$h^{R1}_{it} = BiLSTM(r^{R2}_{it}), \tag{15}$$

$$r^{R1}_{it} = \frac{1}{2}(h^{R1}_{it} + h^{E1}_{it}), \tag{16}$$

$$\tilde{e}_{it} = BiLSTM(r^{R1}_{it}). \tag{17}$$

where h^{Ej} and h^{Rj} are output hidden states of the Bi-LSTM layer j of the encoder and reconstruction decoder respectively (we slightly abuse the notation of eqn. 6 here). Similarly, for the prediction decoder g_P:

$$z''_t = W'' z_t + b'', \tag{18}$$

$$h^{P2}_{it} = BiLSTM(z''_t), \tag{19}$$

$$\tilde{e}_{i,t+1} = BiLSTM(h^{P2}_{it}), \tag{20}$$

3) Time Delay Neural Network (TDNN): Our HAAE network uses LSTM as the core temporal processing unit, however any other sequence modelling architectures could be employed. In this work we also investigate TDNN [19], [21] as a LSTM alternative. Unlike LSTM which uses memory state to model long term dependencies, a TDNN cell instead loops back the output directly as one of its inputs for the next time step. To handle short-term dependency TDNN uses a set of convolution filters spanning across a time window k. Under TDNN, eqn. 3 becomes:

$$h_{it} := TDNN(e_{it}) = \sigma([\boldsymbol{e}_{it}, h_{i-1,t}]W + b). \tag{21}$$

where $\boldsymbol{e}_{it} = [e_{i-k/2,t}, ...e_{it}, ...e_{i+k/2,t}]$. That is, the output at a certain time step is pulled from k neighbouring inputs. Since our time window is symmetrical ($\pm k/2$) the bidirectional setting is no longer necessary. TDNN does not have memory state like LSTM and relies solely on convolutions thus is much faster.

4) Losses: Training our HAAE network is governed by three equally-contributed losses, minimising the overall objective function:

$$M_{(\mathcal{X})} = \underset{\theta_E,\theta_R,\theta_P}{\arg\min}\, \mathcal{L}_R(X_t;\theta_E,\theta_R) + \\ + \mathcal{L}_P(X_t, X_{t+1};\theta_E,\theta_P) + \mathcal{L}_T(X_t;\theta_E). \tag{22}$$

where $\theta_E, \theta_R, \theta_P$ are the weight parameters of the encoder f_R and the two decoders g_R and g_P.

Reconstruction loss, $\mathcal{L}_R$, aims to reconstruct input sequence $\hat{E}_t$ from z_t that approximates E_t. This loss is measured using Mean Square Error (MSE), effectively makes the network an auto-encoder as in [26].

$$\mathcal{L}_R(X_t) = \frac{1}{2}|f_C(X_t) - g_R(f_R(f_C(X_t)))|_2^2 \\ = \frac{1}{2}|E_t - \hat{E}_t|_2^2. \tag{23}$$

Prediction loss, $\mathcal{L}_P$, aims to predict the next sequence $\hat{E}_{t+1}$ from z_t that approximates E_{t+1} via MSE. While the reconstruction loss ensures the integrity within sequences, the prediction loss provides inter-sequence links which is important for encoding very long videos. For the final sequence $t = T$, this loss term is simply turned off. This work is similar to [16] however they used 3D CNN for spatio-temporal encoding instead of HAAE in our work.

$$\mathcal{L}_P(X_t, X_{t+1}) = \frac{1}{2}|f_C(X_{t+1}) - g_P(f_R(f_C(X_t)))|_2^2 \\ = \frac{1}{2}|E_{t+1} - \hat{E}_{t+1}|_2^2. \tag{24}$$

Triplet loss, $\mathcal{L}_T$, brings together similar sequences (z_a, z_p) while pushing dissimilar sequences (z_a, z_n) such that their difference is below a certain margin:

$$\mathcal{L}_T = \frac{1}{|T|} \sum_{(z_a,z_p,z_n)\in T} \{d(z_a, z_n) - d(z_a, z_p) + m\}_+ \tag{25}$$

where $d(.)$ is the cosine similarity metric and $\{.\}_+$ is the hinge loss function. Here we desire the embedding Z invariant to changes in video formats (*e.g.* codec, container, compression quality) hence the positive sample, X^p_t, is set to be the same video sequence as the original sequence X_t but transcoded differently $z_p = f_R(f_C(X^p_t))$. To generate X^p_t we augment $\mathcal{X}$ using various transcoding combinations (details are described in subsec. sections IV-A and IV-B), resulting in a much larger training set $\mathcal{X}^P$ that also helps to reduce overfitting. To make the learning harder, the negative sample is selected among other sequences having the same encoding format as the anchor $z_n = f_R(f_C(X^n_{t*}))$, $X^n_{t*} \in \mathcal{X}\backslash X_t$. Additionally, in archiving practice it is more desirable to reduce the false negative rate than the false positive, thus we also create negative samples by randomly removing several frames from the anchor (and pad with frames from the next block X_{t+1} in order to maintain the sequence length).

The margin m is fixed $m = 0.2$ in our experiments. The network is trained end-to-end via the Stochastic Gradient Descent optimizer (SGD with momentum, decay learning rate and early stopping following [53]) with InceptionResNetV2 initially pre-trained over the ImageNet corpus [50].

5) Encoding audio content: We propose a similar HAAE network to encode audio, replacing the CNN encoder function f_C with MFCC co-efficients from the audio track of each block X_t. MFCC [54] is chosen because it closely approximates human vocal system's response, which is arguably important in archiving. We later demonstrate that encoding audio is critical in the cases where video contains mostly static scenes e.g. a video account of a meeting or court proceedings. The HAAE bottleneck for audio features is 128-D; we denote this embedding z'_t.

B. Video Hashing, Compression and Tampering Detection

Given a video $\mathcal{X}$, the representation z_t of sub-sequence X_t where $t = [0, T]$ can be obtained by passing X_t through the encoder $f_R(f_C(.))$. Additionally, for each scene we define a threshold value set to tolerate variation due to video transcoding:

$$\epsilon_t = \max_{z^p_t \in Z^p_t} ||z_t - z^p_t||_2 \tag{26}$$

where Z^p_t is the collection of bottleneck representations of the positive clips (*i.e.* transcoded) X^p_t used to form the training

Fig. 2. Representative thumbnails of the long-duration video datasets used in our evaluation. Top: ASSAVID, uneditted 'rushes' footage from the BBC archives of the 1992 Olympics; Middle: OLYMPICS, editted footage from Youtube of the 2012-2016 Olympics; Bottom: TNA, Court hearings — mostly static visuals.

TABLE I
DATASET STATISTICS.

Datasets	Video No.	Total length (hh:mm:ss)	Min/Max/Avg	Format
ASSAVID [56]	21	01:29:11	12s/39m/4m	mpg, mpeg
TNA	7	06:23:54	4m/2h/54m	mp4, flv
OLYMPICS	5	00:48:43	8m/13m/9m	mp4

triplets. ϵ_t should tolerate any video transcoding in future with assumption that future codecs are likely to encoder with higher fidelity than existing methods. The same method is applied to process audio.

The collective representations of all blocks that forms $\mathcal{X}$, along with other meta info such as threshold ϵ_t and clip IDs are stored securely in blockchain as the content hash of video $\mathcal{X}$ (more details in subsec. V). As $\mathcal{X}$ can have an arbitrary length, its content hash could exceed the current blockchain transaction limit. To address this problem, we further compress Z into 256-bit binary using Product Quantisation (PQ) [55]. PQ requires the training of a further model (referred to here as 'PQ Encoder') capable of compressing $Z \in \mathbb{R}^{256}$ to $\zeta \in \mathbb{B}^{256}$ and estimating the L_2 norm between pairs of such feature representations.

At verification time, we determine the integrity of a query video X^* by feeding forward each of its blocks through the video and audio models trained for $\mathcal{X}$, to obtain its TCHs. These hashes of $\mathcal{X}^*$ are compared against the corresponding hashes of $\mathcal{X}$ stored within the blockchain. $\mathcal{X}^*$ is considered tampered if there exists a pair of hashes whose distance greater than the corresponding threshold (ϵ_t). This method is advantageous as it provides a localised indication of which block within the video experienced tampering.

IV. EXPERIMENTS

We describe the datasets used as the benchmarks for our experiments in subsec. IV-A and details of our model settings and training procedure in subsec. IV-B. We then present our ablation studies in subsec. IV-C, the tampering detection results in subsec. IV-D and time complexity in subsec. IV-E.

A. Datasets

In archiving practice, digital videos can be of any topic and arbitrary length – a typical video record of an event can last several hours. Moreover, due to its saving-for-future purpose and the diversity in archiving methodology of different archive states, there is no standard encoding format for these videos. In contrast, most video datasets in the literature are few seconds to several minutes in length and of relatively narrow domain. In this work we evaluate with 3 diverse datasets more representative of an archival context as follows:

ASSAVID [56] are unedited television broadcast videos taken from Barcelona Olympics of 1992. This dataset covers various sports with commentary in the audio track. Videos are encoded using the dated MPEG-2 video and MP2 audio codecs.

OLYMPICS are edited fast-moving Youtube videos taken from Olympics 2012-2016 covering final rounds of competitions in 5 different sports including running, swimming, skating and springboard. The dataset has modern high quality MPEG-4 encoding but often contains fast motions and loud background noise and/or music.

TNA are court-hearing videos released by the United Kingdom National Archives. These videos are encoded using either Sorenson H.263/MP3 or H.264/AAC. This dataset is particularly challenging as the videos contains highly static scenes of courtroom but salient audio track, also several of them last hours. Examples of these datasets are shown in Fig. 2 and its statistics are summarised in Tab. I.

For each video within the three datasets above, we create three test sets: a '*control*' set, a '*temporal tamper*' set and a '*spatial tamper*' set. The '*control*' test set contains 10 reformatted versions of each of the original videos. Specifically, we use the modern video codec H.265 with random frame rate (26/48/60 fps), 12 levels of compression and containers (.mp4, .mkv) to transcode the videos. Audio is also transcoded to a new codec (libmp3lame, aac) with random sampling rate and compression quality. The transcoding is implemented using the open-source *ffmpeg* library. In the worst scenario the transcoded video could be 8 times smaller in file size than the original video. The '*temporal tamper*' test set contains 100 videos generated from each original video by randomly removing a chunk of 1-10 seconds while keeping the same encoding formats. Similarly, the '*spatial tamper*' test set contains 100 videos where frames/audio within random sections of the video (1-10 seconds in length) are replaced by white noise.

B. Training Procedure

Given video clip $\mathcal{X}$ we train two HAAEs independently yielding a pair of models (M, M') for the video and audio modalities. We employed InceptionResnetV2 [50] (S=1536) as the final CNN architecture for f_C (we further discuss this in subsec. IV-C1) and using the embedding provided by the final fully connected/pooling layer as the output to E. To minimise model size we freeze f_C and only train $\{f_R, g_R, g_P\}$. Input frames are resized to fit the CNN input size *i.e.* 299x299 for InceptionResNetV2. Input audio is first converted to a mono stream (when necessary) and normalised prior to which MFCC

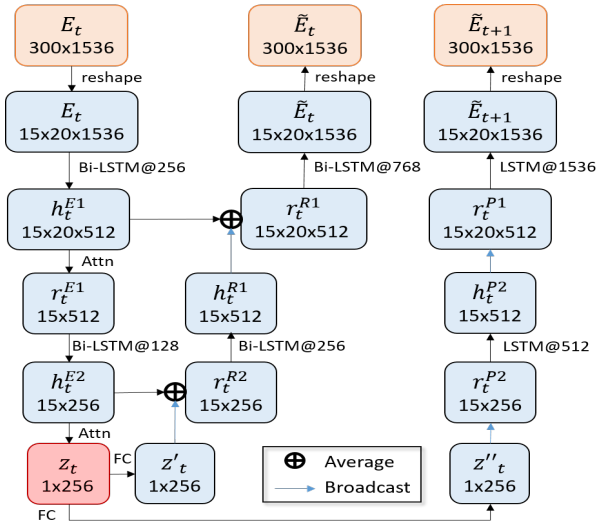


Fig. 3. Details of the encoder and decoders' layers in our HAAE-LSTM network encoding video content. The CNN architecture (omitted in this figure) is InceptionResNetV2 thus E_t has feature dimension $S = 1536$.

TABLE II
SEQUENCE LENGTH BREAKDOWN IN HAAE. N: LENGTH OF THE INPUT SEQUENCE TO THE HAAE NETWORK, WHICH IS SPLIT INTO q SUB-BLOCKS OF LENGTH p EACH ($N = p \times q$) TO BE PROCESSED HIERARCHICALLY – C.F. FIG. 1.

Video length (minutes)	Block size (N)	Layer 1 seq. len. (p)	Layer 2 seq. len. (q)
(0, 2]	150	15	10
(2, 15]	300	20	15
(15, ∞)	600	40	15

features are extracted (S=26). For video content, the HAAE encoder f_R has 256 and 128 LSTM cells in its two layers while the decoders g_R and g_P are configured according to the encoder settings and the feature dimension of the input (see Fig. 3). Note that the prediction decoder g_P has 2x number of LSTM cells compared with the reconstruction decoder g_R to compensate for the lack of bridging and LSTM bidirectional parsing. The block length (N) and sequence length for the two hierarchical layers (p, q) are set according to Tab. II. We set the block size small for short videos to encourage coherence learning between subsequent chunks and maintain sufficient training data (small block size means more blocks available for training). Discussion of these settings are reported in subsec. IV-C4.

For audio content the number of LSTM cells halve to model the much more compact MFCC features. The dimension of the embedding vector z is empirically set to 256-D for video and 128-D for audio prior to PQ. Once the model has been trained we save the encoder part of the network only; the off-chain storage required to verify a video is approximately 50Mb total for audio and video models.

Data augmentation in the form of video reformatting, is used to enrich the training data and form the triplets to develop codec invariance (eqn. 25). For each video we created 50 different transcoded versions using current or older codecs (*e.g.* H.264, H.263p, VP8, ...) at a variety of frame rates from 20fps to 30fps and with a variety of compression/quality parameters spanning typical high and low values for each codec. The transcoder used was *ffmpeg* with compression factor 'crf' an integer number reflecting presets for the codecs provided with that library. Note that these codec sets differ from those natively used in the test set. These transcoded examples form the positive exemplars for the triplet network (subsec. III-A4). The negative exemplars are sampled from the two following sources at equal probabilities: (i) a block semantically different from the anchor and (ii) the anchor block with maximum 5% frames removed or replaced by noise that simulates a tampering attack.

We conducted our experiments using the Python deep-learning library Tensorflow on a machine with 200GB RAM, Intel Xeon 6140 (max 6 cores per job) and a 16GB Quadro GTX 5000 GPU.

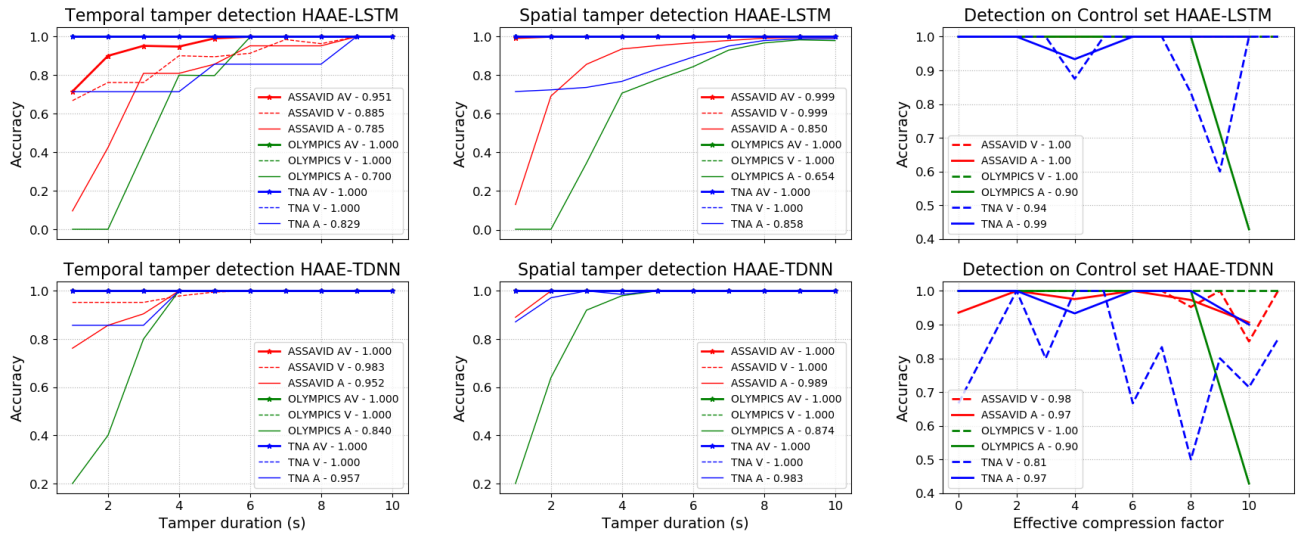
TABLE III
EVALUATING VISUAL TAMPER DETECTION PERFORMANCE FOR VARIOUS CNN ARCHITECTURES. VALUES EXPRESSED ARE ACCURACY RATES FOR *spatial* AND *temporal* TAMPER DETECTION, AND TRUE NEGATIVE RATES FOR *control* AVERAGED ACROSS THE ABLATION TEST SET. HIGHER IS BETTER.

Architecture	Params (M)	Speed (ms)	Temporal	Spatial	Control
NASNet [57]	4.3	35.7	0.833	0.696	0.873
MobileNetV2 [58]	**2.3**	**10.5**	0.849	0.697	0.864
DenseNet [59]	7.0	24.2	0.852	0.699	**0.973**
VGG-16 [60]	134.3	11.2	0.876	0.753	0.909
ResNet50 [17]	23.6	15.6	0.856	0.747	0.913
Incep.ResNetV2 [50]	54.3	46.7	**0.898**	**0.809**	0.955

C. Ablation Studies

We implemented ablation studies for different encoder backbones, choice of loss functions, hyper-parameter tuning and various elements in our HAAE design. We sampled 12 videos from all datasets above to create a single ablation test set. All experiments below are tested against this test set unless otherwise stated.

1) CNN architecture: We tested 6 architectures for the CNN encoder part, including 3 large models and 3 compact ones. The large models are 16-layer VGG [60] (S=1024), 50-layer ResNet [17] (S=2048) and Inception-ResnetV2 [50] (S=1536). The compact models include MobileNetV2 (S=1280), the mobile version of NASNet (S=1056) and the smallest version of DenseNet (aka. DenseNet-121, S=1024). For simplicity and without loss of generalization we use an 1-layer LSTM design for f_R, g_R, g_P. When varying the CNN backbone all other parts of the network are kept the same. Table. III shows the tamper detection accuracy on the *temporal* and *spatial* tamper sets, as well as the true negative rate for the control sets. The model size (total number of parameters excluding the last FC classification layer) and feature extraction speed (measured in ms per frame) are also reported. The overall performance of these models mostly reflect their classification scores on ImageNet. An exception is VGG16 which surprisingly outperforms the 3 compact models. VGG16 is also the runner-up in term of inference speed despite being the largest one (because it creates less intermediate feature-maps during inference). MobileNetV2 enjoys being



(a) Temporal (b) Spatial (c) Control

Fig. 4. Evaluation of HAAE-LSTM (top) and HAAE-TDNN (bottom) on the temporal (a), spatial (b) and control (c) test sets of the three datasets listed in subsec. IV-A.

the most compact and fastest model although sub-optimal accuracy. DenseNet has the most compact output (S=1024) and outperforms the rest on the control set, although its relatively low performance on the tamper sets might indicate the model struggles to fit on data. Overall, the InceptionResNetV2 architecture shows consistent performance on all test sets. It achieves the best scores on the temporal and spatial sets (2% and 5% better than the closest competitors respectively) and second-best on the control set. InceptionResNetV2 is therefore selected as the final CNN backbone in our HAAE design for the next experiments.

2) Losses: We perform ablations on the three terms comprising total loss (eqn. 22) while keeping the network, data and training parameters unchanged. The following baselines are brought to comparison:

- **AE** [26] – a standard autoencoder with only the reconstruction loss $\mathcal{L}_R$ being enabled.
- **AE+P** [16] – an autoencoder with a prediction branch. $\mathcal{L}_R$ and $\mathcal{L}_P$ are enabled.
- **Triplet AE+P** [4] – a fully shared triplet network where each of the anchor/positive/negative branches is an autoencoder with a prediction decoder as a tendril. All loss terms in eqn. 22 are enabled.

The contribution of each loss to tamper-detection capability is reflected in Table. IV. The reconstruction loss alone (*i.e.* the AE model) has the best score on the control set since its sole objective is to reconstruct the input sequences. The lack of learning coherence between adjacent sequences causes it to under-perform detection on both tamper sets. In contrast, the prediction loss (AE+P) helps to learn an embedding more sensitive to tampering and the triplet loss (Triplet AE+P) balances the performance on all three test sets. This demonstrates the importance of the three loss terms of eqn. 22 in the detection of tampered videos.

3) HAAE: We experimented with different designs of our HAAE architecture by disabling one or more of its features as well as changing the core layers. The following algorithms are brought to comparison:

- **SimpleRNN** – a simple RNN using only FC layers where the output is to be fed back to input. SimpleRNN is similar to TDNN in the sense that it does not hold any memory state however SimpleRNN does not have the notation of time window like TDNN.
- **Gated Recurrent Unit (GRU)** [22] – stores both long and short term memory in a single hidden state. GRU is a less complex but faster version of LSTM.

TABLE IV
ABLATION EXPERIMENT EXPLORING TERM INFLUENCE ON TOTAL LOSS (EQN. 22): $\mathcal{L}_R$ (RECONSTRUCTION), $\mathcal{L}_P$ (PREDICTION) AND $\mathcal{L}_T$ (TRIPLET). SOLID DOT INDICATES INCORPORATION OF LOSS TERM. VALUES EXPRESSED ARE ACCURACY RATES FOR *spatial* AND *temporal* TAMPER DETECTION, AND TRUE NEGATIVE RATES FOR *control* AVERAGED ACROSS THE ABLATION TEST SET. HIGHER IS BETTER.

Method	$\mathcal{L}_R$	$\mathcal{L}_P$	$\mathcal{L}_T$	Temporal	Spatial	Control
AE [26]	•			0.884	0.749	**0.968**
AE+P [16]	•	•		0.896	0.776	0.945
Triplet AE+P [4]	•	•	•	**0.898**	**0.809**	0.955

TABLE V
ABLATION EXPERIMENTS FOR DIFFERENT HAAE SETTINGS. VALUES EXPRESSED ARE ACCURACY RATES FOR *spatial* AND *temporal* TAMPER DETECTION, AND TRUE NEGATIVE RATES FOR *control* AVERAGED ACROSS THE ABLATION TEST SET. HIGHER IS BETTER.

Architecture		Temporal	Spatial	Control
GRU [22]		0.825	0.690	0.945
SimpleRNN		0.900	0.793	0.927
LSTM	Stacked LSTM [24]	0.842	0.787	0.891
	1-layer LSTM/woT [4]	0.898	0.809	0.955
	HAE-LSTM/woT	0.956	1.00	0.954
	HAAE-LSTM/woT	0.960	0.999	**0.965**
	HAAE-LSTM (final)	0.988	**1.00**	0.964
TDNN	HAE-TDNN/stride [20]	0.838	0.767	0.920
	HAAE-TDNN/woT	0.981	1.00	0.963
	HAAE-TDNN (final)	**1.00**	**1.00**	0.963

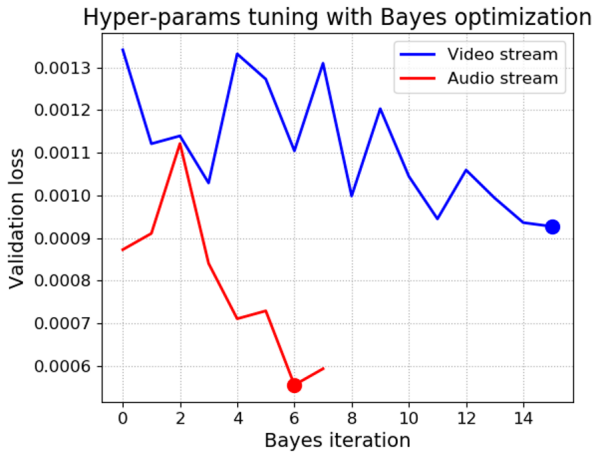
- **Stacked LSTM** [24] – a 2-layer LSTM network where outputs of every time steps in the first layer are the corresponding inputs for the next layers.
- **1-layer LSTM/woT** [4] – the same **Triplet AE+P** model reported in Tab. IV, consisting just a single layer LSTM with all loss terms functioning. Note the notation **woT** indicates 'without tampering data' seen by the model during training.
- **HAAE-LSTM** – the proposed approach described in subsec. III-A. We also experimented with 2 more variants. First, **HAE-LSTM/woT** is the hierarchical network without attention module. Second, **HAAE-LSTM/woT** is the proposed approach without tampering data seen during the training.
- **HAAE-TDNN** – the same HAAE network with TDNN replacing LSTM. We tested 3 variants of this approach. The full version and **HAAE-TDNN/woT** mirror the HAAE-LSTM configurations above. The 3rd configuration, **HAE-TDNN/stride** is inspired from [20] where convolution stride is employed to compress sequence length between layers instead of using attention mechanism.

The results are depicted in Tab. V. We made four observations. First, LSTM and TDNN based approaches are generally better than GRU and SimpleRNN. Second, the hierarchical structure contributes significantly to the performance gain, as in the case of HAE-LSTM versus 1-layer LSTM. This indicates the benefits of modelling short sub-sequences within a long sequence in a hierarchical order even if such order may not exist in the data. Third, integrating the attention mechanism also benefits the performance significantly, as in HAAE-LSTM versus HAE-LSTM and especially HAAE-TDNN versus HAE-TDNN/stride. Finally, the tamper detection accuracy could be improved further by supplying synthesised tamper data as negative exemplars to the model during training.

Since the TDNN-based approaches have demonstrated impressive performance on the ablation tests we decided to evaluate it further on the full datasets in subsec. IV-D, along with LSTM-based HAAE.

4) Tuning hyper-parameters: Selecting the right values of hyper-parameters is an important task. During model training we employed adaptive learning rate (specifically, the learning rate is step-decayed if the loss does not improve) and early stopping (if the learning rate reaches its minimum and the loss still does not improve after certain epochs the training is stopped), hence the learning rate and number of epochs are exempted from hyper-parameter tuning. Other important parameters in HAAE include number of LSTM cells in each layer and the layer's sequence length (denoted as p and q in subsec. III-A). Inspired from the work of [61], [62] , we employed a Bayesian Optimization method based on Gaussian Process to find optimal values of these parameters[1]. Fig. 5 shows the tuning results on the video and audio streams of a sample video. Having much smaller feature dimension, the tuning of the audio data often stops after less than 10 Bayes iterations while the video stream took much longer. Also we observed that the optimal values vary greatly across

[1]This method is implemented in the open-source GPyOpt Python library.



	Layer 1 #cells	Layer 2 #cells	Layer 1 seq. len.	Layer 2 seq. len.
Video	281	145	14	10
Audio	201	101	13	11

Fig. 5. Tuning hyper-parameters of HAAE using Bayes optimization on a sample video. Optimal values are shown in the table below. The objective function is validation loss, which is the averaged total loss over a transcoded video set aside from training.

different videos. Since a separate HAAE model is trained for each video we could ideally run Bayesian Optimization on every training. However this is a time-consuming process and quickly becomes impractical on long videos. We therefore opt to a fixed hyper-param setting as described in subsec. IV-B and Tab. II, which is the closest to our observations of the optimal values on the ablation set.

D. Tamper Detection

We evaluated the performance of tamper detection on all 3 full datasets for a variety of tamper lengths. Fig. 4 (a-b) shows the trend of tamper length to detection accuracy for the proposed system and also on an ablated version of the system reliant solely on audio and solely on video modalities. Both *temporal* and *spatial* tamper sets have a similar trend: models are consistently effective at detecting tampered segments of around 2 seconds or more. This is excellent sensitivity given the total length of the individual videos (c.f. Tab. I). The TNA and ASSAVID datasets have the most clear audio thus achieves great audio scores on both test sets. In contrast, the lowest audio score is reported on the OLYMPICS dataset which contains mostly music (sub-optimal for MFCC) or loud background noise. Both the TNA and OLYMPICS have high visual quality that leads to well performant video models. It also demonstrates the robustness of our approach against video motion – whether the videos are static like the TNA or contain fast-moving objects like the OLYMPICS. On the other hand, low visual quality and older codecs degrade the performance on the ASSAVID's video streams. Fig. 6 illustrates that the decay in image quality during video transcoding and compression on the ASSAVID is more severe than the other two datasets. This results in a more scatter embedding for those transcoded data thus larger tampering threshold (eqn. 26). Consequently the ASSAVID

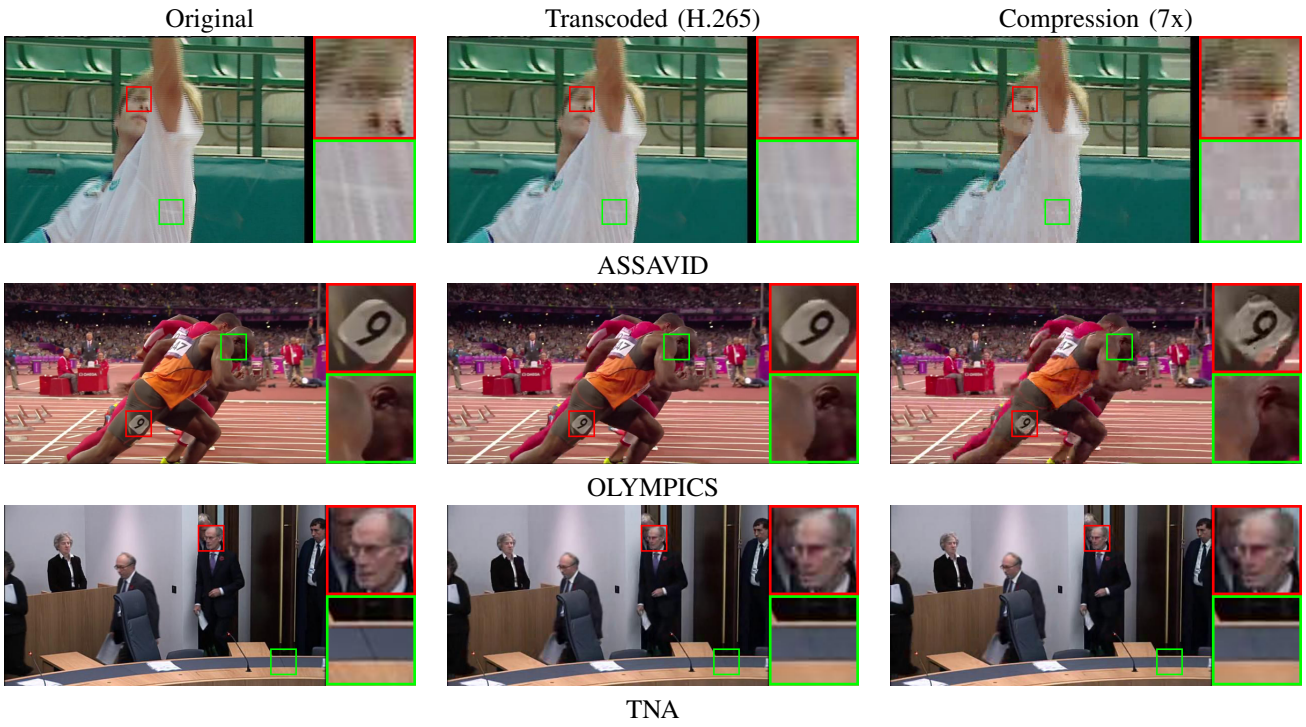


Fig. 6. Artefacts on video frames after transcoding and compression using ffmpeg. Note the compression ratio is controlled via the *crf* parameter (higher crf value means more compression, smaller footprint but lower quality). Having videos encoded with an old codec, the ASSAVID dataset experiences the heaviest degradation in image quality.

models become less sensitive to tampering, especially with short tamper duration.

Fig. 4 (a-b) also demonstrates the benefits of modelling both video and audio modalities together. As with the OLYMPICS dataset, a perfect tamper detection score is achieved thanks to the video model compensating the difficult audio model. For ASSAVID, the two modality models compliment each other to deliver a significant performance boost on both tampering test sets. Additionally, the TDNN-based approach (bottom row) demonstrates better performance than LSTM (top row). The LSTM-based approach has comparable performance when aggregating results from both audio and video streams. Furthermore, HAAE-LSTM has better performance on the control set, as shown in Fig. 4 (c). Here the true negative detection rate is plotted against the compression factor of the transcoded videos. In overall, accuracy is consistently high for video downgrade, with no observable trend that severely compressed videos are incorrectly flagged as tampered. This is the results of data augmentation during training – the models are trained to be robust against downgrade in audio-visual quality (subsec. III-A4 and IV-A). The tampering detection accuracy on the audio stream remains high except for the OLYMPICS dataset which starts dropping at $q = 8$. This is understandable considering the noisy audio signal present in the OLYMPICS set. The performance on OLYMPICS-A could be improved further by filtering noise out of the audio signal and retaining only human speech as a pre-processing step but it is beyond the scope of this work.

We note the low performance of HAAE-TDNN compared with HAAE-LSTM on the TNA set in Fig. 4 (c). We hypotheses that TDNN struggles to model large amount of data, as the TNA set contains mostly long videos (c.f. Tab. I). To confirm this hypothesis we plot the performance statistics across all datasets grouping videos by length (Fig. 7). While the performance on the two tamper test sets is shown not affected by video length, the detection rate on the control set *i.e.* the ability of the system to reflect true negatives (avoid false detections) gradually decreases, with faster drop for the TDNN-based approach. The standard deviation in measurement on the control set (illustrated by the whiskers on top of the accuracy bars in Fig. 7) also increases, which is the result of our system outputting binary decisions. The drop in performance on the control set can be explained in 2 ways. First, for extremely long videos there is a greater chance that two duplicated or near duplicated frame blocks exist, particularly from highly static events recorded from stationary cameras such as the court hearing events in the TNA dataset. Second, it is statistically justifiable – an hour-length video consists of many frame blocks in which one failing block will label the whole video as tampered. Nevertheless, approximately 10% of the videos have length greater than 1 hour, yet for HAAE-LSTM 90% of the control set are detected correctly and more importantly the tamper videos are all successfully detected. HAAE-LSTM is also shown more robust in modelling long videos compared with HAAE-TDNN.

Table. VI summarises the performance of our tamper detection system in term of Precision (positive predictive value), Recall (sensitivity), False Detection Rate (FDR), overall Accuracy (ACC) and F1 score (after [63]). Similar to our observation in Fig. 4 the ASSAVID dataset with dated MPEG-2 codec and low video quality results in large tampering threshold ϵ_t (eqn. 26) thus less sensitive to tampering. The

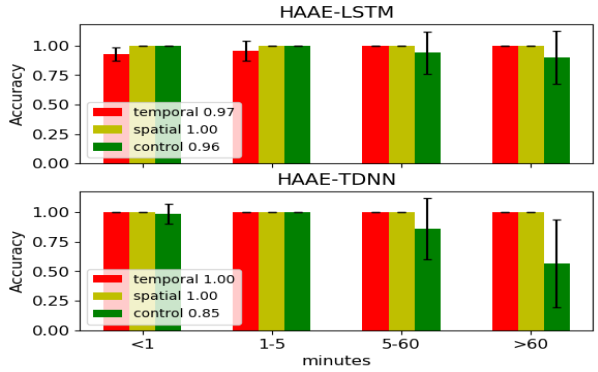


Fig. 7. Overall accuracy vs. video length averages across all three test sets for HAAE-LSTM and HAAE-TDNN. Values expressed are accuracy rates for *spatial* and *temporal* tamper detection, and true negative rates for *control* averaged across all test sets partitioned by length.

TABLE VI
OVERALL TAMPER DETECTION PERFORMANCE OF THE 3 DATASETS IN TERM OF PRECISION (P), RECALL (R), FALSE DETECTION RATE (FDR), ACCURACY (ACC) AND F1 SCORE. HERE, A TRUE POSITIVE DENOTES A TAMPERED VIDEO BEING CORRECTLY DETECTED AS TAMPERED. HIGHER IS BETTER FOR P/R/FDR/F1. LOWER IS BETTER FOR FDR.

Methods	Datasets	P	R	FDR	ACC	F1
HAAE-LSTM	ASSAVID	1.00	0.9733	0.00	0.9746	0.9865
	OlYMPICS	0.9960	1.00	0.0040	0.9962	0.9980
	TNA	0.9964	1.00	0.0036	0.9966	0.9982
HAAE-TDNN	ASSAVID	0.9979	1.00	0.0021	0.9980	0.9989
	OlYMPICS	0.9960	1.00	0.0040	0.9962	0.9980
	TNA	0.9894	1.00	0.0106	0.9898	0.9947

more modern MPEG-4 encoded OLYMPICS dataset has a better balance between precision and recall with near-perfect overall performance. The HAAE-TDNN is efficient on the short ASSAVID videos but the HAAE-LSTM proves more robust on the longer TNA set.

E. Time complexity

We report the time complexity of the training and test phases for our proposed HAAE-LSTM and HAAE-TDNN approaches. Fig. 8 (top) illustrates the time spent at different training stages. A typical training time on a 3-minute video is 2 hours. For a 90 minute video it takes 27.5 hours in total for HAAE-LSTM and 23 hours for HAAE-TDNN. The most time consuming train step - CNN feature extraction - accounts for $\sim 75\%$ of total training time. Note that this time is mainly spent on decoding and parsing video frames rather than the actual CNN model inference, although this could be improved with more multiprocessing resources. Training the HAAE model is also time-consuming with more time spent on the LSTM-based approach. Fig. 8 (bottom) compares the time efficiency of HAAE-LSTM and HAAE-TDNN in training (excluding all common train steps) and test. HAAE-TDNN is much faster than HAAE-LSTM at training although the difference at test time is negligible. Fig. 8 (bottom) also shows the linear relationship between train/test time and video length (note: the time plots indicates exponential curves but the video-length axis is in logarithmic scale).

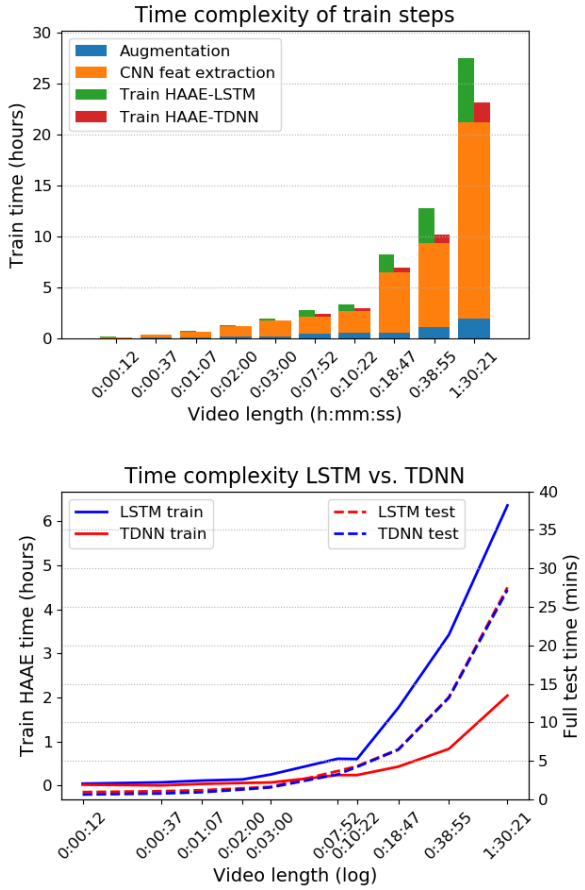


Fig. 8. Time complexity of our TCH system versus video length. (Top) time taken at different train steps. Note: HAAE-LSTM and HAAE-TDNN have identical augmentation and CNN feature extraction steps. (Bottom) Speed gain when replacing LSTM layers in HAAE with TDNN.

V. ARCHANGEL: SAFEGUARDING DIGITAL ARCHIVES WITH AI AND BLOCKCHAIN

We present ARCHANGEL – a proof-of-authority (PoA, permissioned) blockchain service as a mechanism for assuring provenance of digital video stored off-chain, within the public National Archives of several nation states. In this section we describe the ARCHANGEL architecture in subsec. V-A, the statistic of our trial studies in subsec. V-B and the current limitations in subsec. V-C.

A. The ARCHANGEL System

The core architecture of ARCHANGEL is a DLT network implemented via Ethereum [64] and maintained by the independently participating archives, each of which runs a sealer node in order to provide consensus by majority on data stored on-chain. Fig. 9 illustrates the architecture of the system. ARCHANGEL is implemented on a compute infrastructure run locally at each participating archive. The system has been designed with archival practices in mind, utilising data packages conforming to the Open Archival Information System (OAIS) [65], and as such the archivist user must provide metadata for an archive data package named Submission Information Package (SIP) that the video files are part of. A web-app run locally inspects each SIP as it

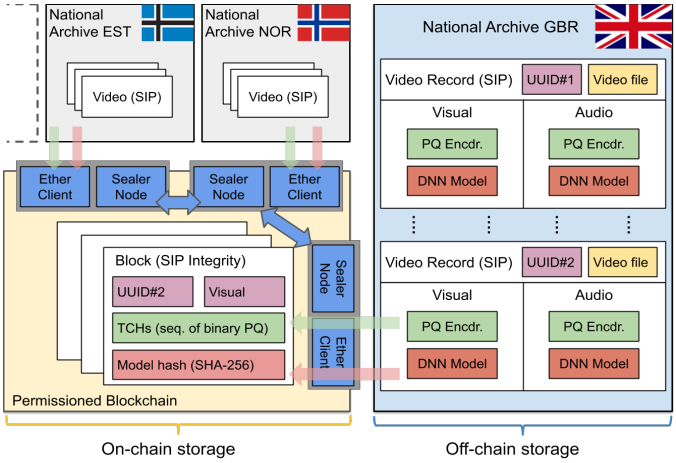


Fig. 9. ARCHANGEL Architecture. Multiple archives maintain a PoA blockchain (yellow) storing hash data that can verify the integrity of video records (video files and their metadata; sub-sec. V) held off-chain, within the archives (blue). Universal unique identifiers (UUID) link on- and off-chain data. Two kinds of hash data are held on-chain: 1) A temporal content hash (TCH) protecting the audio-visual content from tampering, computed by passing sub-sequences of the video through a deep neural network (DNN) model to yield a sequences of short binary (PQ [55]) codes (green); 2) A binary hash (SHA-256) of the DNNs and PQ encoders that guards against tampering with the TCH computation (red).

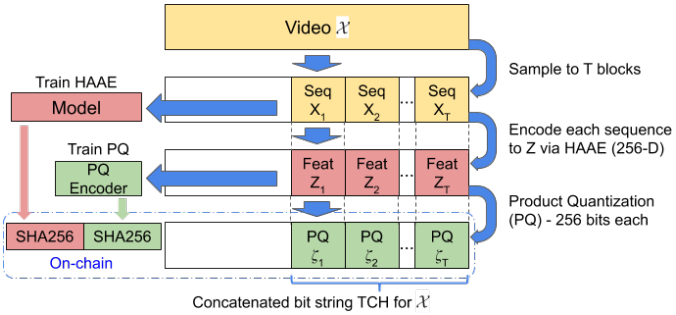


Fig. 10. Schematic of TCH Computation. Videos are split into blocks which are each independently hashed to a TCH. The TCH is the concatenation of PQ codes that are derived from RNN bottleneck features (z_t). Each code hashes a block of N frames. Encoding pipeline shown for visual cues only; audio is similarly encoded.

is submitted, where it is verified to be a suitable file type, and stored within the locally held archive (database) for processing by a background daemon. The daemon trains the video/audio model pair and the PQ encoder on a secure high performance compute (HPC) cluster with nodes comprising four NVidia 1080Ti GPU cards. The time taken to train the model is typically several hours, although subsequent TCH computation takes less than one minute. These timescales are acceptable given the one-off nature of the hashing process for a video record likely to remain in the archive for years or decades.

Given an input video $\mathcal{X}$, hashing each block within $\mathcal{X}$ yields its TCH; a sequence of binary hashes derived from the visual content: $\{X_1, X_2, ..., X_T\} \mapsto \zeta = \{\zeta_1, \zeta_2, ..., \zeta_T\}$ and similarly a sequence of audio hashes $\zeta' = \{\zeta'_1, \zeta'_2, ..., \zeta'_T\}$. These hashes are stored immutably on the blockchain alongside a universally unique identifier (UUID) for the clip. To safeguard the integrity of the verification process, it is necessary to also store a hash of the video/audio models and PQ models as well as its corresponding threshold values ϵ_t also on the blockchain (Fig. 10). Without loss of generality, we compute the model

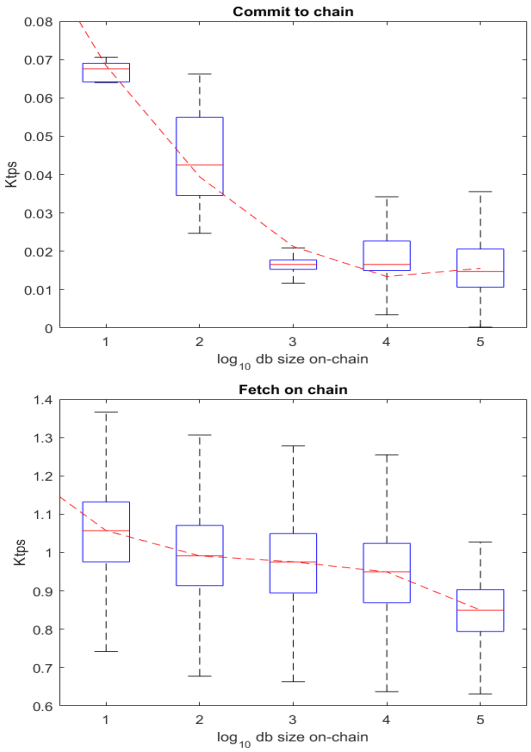


Fig. 11. Performance scalability of smart contract transactions on our proof-of-authority Ethereum implementation. Performance for fetching (top) and commiting (bottom) video integrity records to the on-chain storage. Measured for db sizes from $10^1 - 10^5$ reported as Ktps ($\times 10^3$ transactions per second).

hashes via SHA-256. Use of such a bit-wise hash is valid for the model (but not the video) since the model (which is a set of filter weights) remains constant but the video may be transcoded during its lifetime within the archive. The proposed system requires that the models be stored off-chain due to storage size limitations (32Kb per transaction in our Ethereum implementation), and the potential for reversibility attack on the model itself which might otherwise apply generative methods [66] to enable partial disclosure of timed-release or non-public archival records. Such reversals are not feasible for the 256 bit PQ hashes stored on the chain, particularly given the off-chain storage of the PQ and DNN models.

The TCHs, threshold values and SHA-256 hashes of the video/audio/PQ models necessary for the integrity verification of video $\mathcal{X}$ are stored inside a SIP which is then submitted as a record to the blockchain by way of a smart contract transaction which manages the storage and access of data. In order to submit data the user must be given access via the smart contract, which the application interfaces (APIs) then use to manage access to the submission functionality. Once the transaction is processed, the UUIDs for the SIP and the video files can be used to access data stored on the chain for subsequent verification. For reasons of sustainability and scalability of the system, we opted on a proof-of-authority (PoA) consensus model. Under such a model, there is no need for computationally expensive proof of work mining. Rather, blocks are sealed at regular intervals by majority consensus of the clique of nodes pre-authorised to. In our trial deployment over a small number of archives, access keys were pre-granted

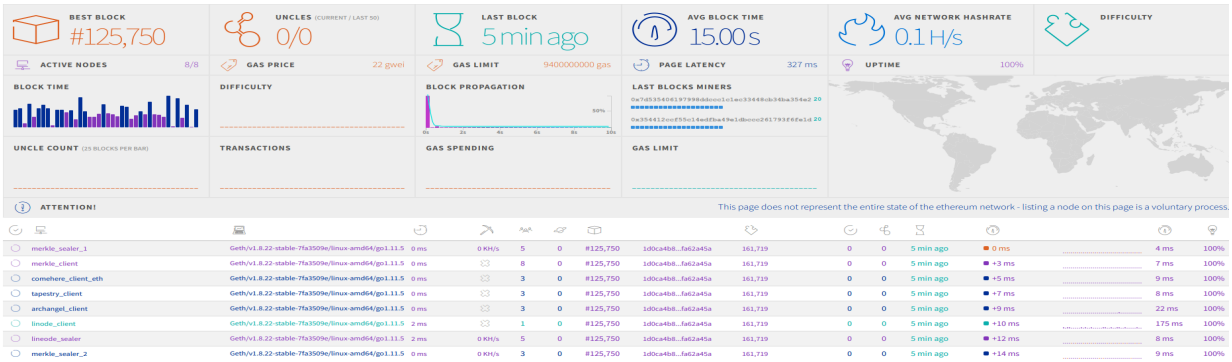


Fig. 12. Ethereum dashboard snapshot of the ARCHANGEL system during early stages of deployment, showing real-time statistics about the state of the network nodes, blocks, and transactions. The active nodes include 5 client nodes and 3 sealing nodes which later expand to 15 nodes in total (National Archives UK x 2, Australia x 1, Estonia x 1, Norway x 1, USA x 1; also ODI x 1, UoS x 8). To ensure a smooth trial the University of Surrey maintained a 50% + 1 share of the nodes, so that new addresses could be easily added without coordination of other parties. In a real-world system all parties would have the same number of nodes, and a consensus from all parties would be required.

and distributed by us. In a larger deployment (or to scale this trial deployment) the PoA model could be used to grant access via majority vote to new archives or memory institutions wishing to join the infrastructure.

B. Blockchain statistics

ARCHANGEL was first setup at the University of Surrey (UoS) and the Open Data Institute (ODI). We conducted a trial study initially at the National Archives of three nations, namely the United Kingdom, Norway and Estonia; later also include the United State of America (USA) and Australia. The smart contracts implementing APIs for the fetch (*i.e.* search for and read a record) and write (commit new record) are deployed on an Ethereum PoA network with *geth* used for client API access and for block sealing at each of the 5 participating archives. A further four block sealing nodes were present on the network for debug and control purposes. For purposes of evaluation the network was run privately between the nodes and genesis block configured with a 15 second default block sealing rate. The sealing rate limits the write (not fetch) speed of the network, so that worst-case transaction processing could be up to 15 seconds. The smart contract implemented fetch functionality via the *view* functions provided by the Solidity programming language, which only read and do not mutate the state, hence not requiring a transaction to be processed.

To test the smart contract we devised a test of writing a number of records to the store, measuring the time taken to create and submit a transaction, and then reading a record a number of times to measure the time taken to read from the store. When writing the records we submitted them in batches of 1000 to ensure that the transactions would be accepted by the network. The transaction creation/submission performance is presented in Fig. 11 (top) and do not include the (up to) 15 second block sealing overhead since this is a constant dependent on the time at which the commit transaction is posted. The transactions were submitted in powers of 10, and then the read performance was measured (Fig. 11 (bottom)). A dashboard snapshot of the Ethereum network is included in Fig. 12.

C. Limitations

Currently ARCHANGEL is limited by Ethereum (rather than via design) to block sizes of 32Kb. Our TCHs for a given video are variable length bit sequences requiring 576 bits per minute to encode audio and video on-chain (256-bit compressed TCH plus 32-bit threshold value for each video/audio minute), plus offsets for model hashes and other meta data in SIP. This effectively limiting the size of digital videos we can handle to 4 hours maximum duration. ARCHANGEL also has a relatively high overhead in off-chain storage in that a model of 50Mb must be stored per video; however this model size is frequently dwarfed by the video itself which may be gigabytes in length and presented minimal overhead in the context of the petabyte capacities of archival data stores. One interesting question is how to archive such models; currently these are simply Tensorflow model snapshots but no open standards exist for serializing DNN models for archival purposes. As deep learning technologies mature, and become further ingrained within autonomous decision making *e.g.* by governments, it will become increasingly important for the community to devise such open standards to archive such models.

VI. Conclusion

We have proposed HAAE – a novel self-supervised content hashing network for arbitrary-length archival videos. We train a lightweight model for each of the audio/video content streams for an individual video, modelling the temporal coherence between spatial features using a customised autoencoder with hierarchical structure and attention mechanism. We have demonstrated our HAAE effective in modelling videos lasting from few seconds to hours on 3 archival-representative benchmarks. We report our LSTM-based HAAE efficient in modelling long videos while the short videos may benefit from the simpler HAAE-TDNN version. We show our system robust against format shift (changes in the video codecs) and motion (static or fast moving scene); also be able to detect frame dropout/truncation or corruption due to either accidental bulk transcoding errors or direct attacks.

Our HAAE system is integrated in ARCHANGEL – a PoA blockchain-based service for ensuring the integrity of digital video across multiple archives. A key contribution of ARCHANGEL is the fusion of computer vision and blockchain to immutably store temporal content hashes (TCHs) of video into a private Ethereum distributed ledger network, overcoming the limitations of bit-level hashes like SHA-256/MD5. The hashes could be used to verify the integrity of the original archival video, should it be released in future, in any new formats. ARCHANGEL has been trialled across the national government archives of 5 sovereign nations.

REFERENCES

[1] V. Johnson and D. Thomas, "Interfaces with the past...present and future? scale and scope: The implications of size and structure for the digital archive of tomorrow," in *Proc. Digital Heritage Conf.*, 2013.

[2] M. Jacobs and J. Probell, "A brief history of video coding," *ARC International*, 2007.

[3] H. Gilbert and H. Handschuh, "Security analysis of sha-256 and sisters," in *Proc. Selected Areas in Cryptography (SAC)*, 2003.

[4] T. Bui, D. Cooper, J. Collomosse, M. Bell, A. Green, J. Sheridan, J. Higgins, A. Das, J. Keller, O. Thereaux *et al.*, "Archangel: Tamper-proofing video archives using temporal content hashes on the blockchain," in *Proc. CVPR WS*, 2019, pp. 0–0.

[5] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton Univ. Press, 2016.

[6] B. Coskun, B. Sankur, and N. Memon, "Spatio–temporal transform based video hashing," *IEEE Trans. MM*, vol. 8, no. 6, pp. 1190–1208, 2006.

[7] F. Khelifi and A. Bouridane, "Perceptual video hashing for content identification and authentication," *IEEE Trans. Circuits and Systems for Video Tech.*, vol. 29, no. 1, 2017.

[8] L. Cao, Z. Li, Y. Mu, and S.-F. Chang, "Submodular video hashing: a unified framework towards video pooling and indexing," in *Proc. ACM Multimedia*, 2012.

[9] G. Ye, D. Liu, J. Wang, and S.-F. Chang, "Large-scale video hashing via structure learning," in *Proc. ICCV*, 2013.

[10] J. Song, Y. Yang, Z. Huang, H. T. Shen, and J. Luo, "Effective multiple feature hashing for large-scale near-duplicate video retrieval," *IEEE Trans. MM*, vol. 15, no. 8, pp. 1997–2008, 2013.

[11] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Proc. ICCV*, 2003.

[12] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman, "Total recall: Automatic query expansion with a generative feature model for object retrieval," in *Proc. CVPR*, 2007.

[13] D. Guo, J. Hong, B. Luo, Q. Yan, and Z. Niu, "Multi-modal representation learning for short video understanding and recommendation," in *Proc. ICMEW*. IEEE, 2019, pp. 687–690.

[14] Z. Xu, Y. Yang, and A. G. Hauptmann, "A discriminative cnn video representation for event detection," in *Proc. CVPR*, 2015.

[15] V. E. Liong, J. Lu, Y.-P. Tan, and J. Zhou, "Deep video hashing," *IEEE Trans. MM*, vol. 19, no. 6, pp. 1209–1219, 2016.

[16] D. Kim, D. Cho, and I. S. Kweon, "Self-supervised video representation learning with space-time cubic puzzles," *arXiv preprint arXiv:1811.09795*, 2018.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.

[18] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *Proc. NeurIPS WS*, 2014.

[19] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Signal Processing*, vol. 37, no. 3, pp. 328–339, 1989.

[20] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. INTERSPEECH*, 2015.

[21] M. Kaselimi, E. Protopapadakis, A. Voulodimos, N. Doulamis, and A. Doulamis, "Multi-channel recurrent convolutional neural networks for energy disaggregation," *IEEE Access*, vol. 7, pp. 81 047–81 056, 2019.

[22] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[23] Y. Gu, C. Ma, and J. Yang, "Supervised recurrent hashing for large scale video retrieval," in *Proc. ACM Multimedia*, 2016.

[24] G. Wu, L. Liu, Y. Guo, G. Ding, J. Han, J. Shen, and L. Shao, "Unsupervised deep video hashing with balanced rotation," in *Proc. IJCAI*, 2017.

[25] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *Proc. ICML*, 2015, pp. 843–852.

[26] H. Zhang, M. Wang, R. Hong, and T.-S. Chua, "Play and rewind: Optimizing binary representations of videos by self-supervised temporal hashing," in *Proc. ACM Multimedia*. ACM, 2016, pp. 781–790.

[27] J. Song, H. Zhang, X. Li, L. Gao, M. Wang, and R. Hong, "Self-supervised video hashing with hierarchical binary auto-encoder," *IEEE Trans. Image Processing (TIP)*, 2018.

[28] Y. Bin, Y. Yang, F. Shen, X. Xu, and H. T. Shen, "Bidirectional long-short term memory for video description," in *Proc. ACM Multimedia*. ACM, 2016, pp. 436–440.

[29] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Proc. NeurIPS*, 2017.

[30] Y. Liu and T. Huang, "Exposing video inter-frame forgery by zernike opponent chromaticity moments and coarseness analysis," *Multimedia Systems*, vol. 23, no. 2, pp. 223–238, 2017.

[31] A. V. Subramanyam and S. Emmanuel, "Video forgery detection using hog features and compression properties," in *Proc. IEEE MMSP WS*. IEEE, 2012, pp. 89–94.

[32] P. Bestagini, S. Milani, M. Tagliasacchi, and S. Tubaro, "Local tampering detection in video sequences," in *Proc. IEEE MMSP WS*. IEEE, 2013, pp. 488–493.

[33] K. Sitara and B. M. Mehtre, "Digital video tampering detection: an overview of passive techniques," *Digital Investigation*, vol. 18, pp. 8–22, 2016.

[34] C. Long, A. Basharat, A. Hoogs, P. Singh, H. Farid, A. Muntakim Rafi, U. Kamal, R. Hoque, A. Abrar, S. Das *et al.*, "A coarse-to-fine deep convolutional neural network framework for frame duplication detection and localization in forged videos," in *Proc. CVPR WS*, 2019, pp. 1–10.

[35] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan, "Recurrent convolutional strategies for face manipulation detection in videos," *Interfaces (GUI)*, vol. 3, p. 1, 2019.

[36] D. D'Avino, D. Cozzolino, G. Poggi, and L. Verdoliva, "Autoencoder with recurrent neural networks for video forgery detection," *Electronic Imaging*, vol. 2017, no. 7, pp. 92–99, 2017.

[37] Y. Li, M.-C. Chang, and S. Lyu, "In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking," *arXiv preprint arXiv:1806.02877*, 2018.

[38] W. Sultani, C. Chen, and M. Shah, "Real-world anomaly detection in surveillance videos," in *Proc. CVPR*, 2018, pp. 6479–6488.

[39] C. Lu, J. Shi, W. Wang, and J. Jia, "Fast abnormal event detection," *IJCV*, vol. 127, no. 8, pp. 993–1011, 2019.

[40] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014.

[41] M. Sharifi, "Patent US9367887: Multi-channel audio fingerprinting," Google Inc., Tech. Rep., 2013.

[42] S. Baluja and M. Covell, "Waveprint: Efficient wavelet-based audio fingerprinting," *Pattern Recognition*, vol. 41, no. 11, 2008.

[43] M. Senoussaoui and P. Dumouchei, "A study of the cosine distance-based mean shift for telephone speech diarization," *IEEE Trans. Audio Speech and Language Processing*, 2014.

[44] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," bitcoin.org, Tech. Rep., 2008.

[45] M. Walport, "Distributed ledger technologies: Beyond blockchain, the blackett review." UK Government Office for Science, Tech. Rep., 2016.

[46] C. Holmes, "Distributed ledger technologies for public good: leadership, collaboation and innovation," UK Government Office for Science, Tech. Rep., 2018.

[47] J. Collomosse, T. Bui, A. Brown, J. Sheridan, A. Green, M. Bell, J. Fawcett, J. Higgins, and O. Thereaux, "ARCHANGEL: Trusted archives of digital public documents," in *Proc. ACM Doc.Eng*, 2018.

[48] V. L. Lemieux, "Blockchain technology for recordkeeping: Help or hype?" U. British Columbia, Tech. Rep., 2016.

[49] B. Gipp, J. Kosti, and C. Breitinger, "Securing video integrity using decentralized trusted timetamping on the bitcoin blockchain," in *Proc. MCIS*, 2016.

[50] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proc. AAAI Artificial Intelligence*, 2017.
[51] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proc. HLT-NAACL*, 2016, pp. 1480–1489.
[52] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. MICCAI*. Springer, 2015, pp. 234–241.
[53] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, "The marginal value of adaptive gradient methods in machine learning," in *Proc. NeurIPS*, 2017, pp. 4148–4158.
[54] M. Sahidullah and G. Saha, "Design, analysis and experimental evaluation of block based transformation in mfcc computation for speaker recognition," *Speech Communication*, vol. 54, no. 4, pp. 543–565, 2012.
[55] H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation," in *Proc. CVPR*, 2010.
[56] K. Messer, W. Christmas, and J. Kittler, "Automatic sports classification," in *Object recognition supported by user interaction for service robots*, vol. 2. IEEE, 2002, pp. 1005–1008.
[57] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. CVPR*, 2018, pp. 8697–8710.
[58] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proc. CVPR*, 2018, pp. 4510–4520.
[59] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. CVPR*, 2017, pp. 4700–4708.
[60] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[61] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on ML*. Springer, 2003, pp. 63–71.
[62] M. Kaselimi, N. Doulamis, A. Doulamis, A. Voulodimos, and E. Protopapadakis, "Bayesian-optimized bidirectional lstm regression model for non-intrusive load monitoring," in *Proc. ICASSP*. IEEE, 2019, pp. 2747–2751.
[63] J. I. Olszewska, "Designing transparent and autonomous intelligent vision systems," in *Proc. ICAART*, 2019, pp. 850–856.
[64] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, pp. 1–32, 2014.
[65] ISO14721:2012, "Open archival information system OAIS," International Standards Organisation, Tech. Rep., 2012.
[66] N. Carlini, C. Liu, J. Kos, Ú. Erlingsson, and D. Song, "The secret sharer: Measuring unintended neural network memorization & extracting secrets," *arXiv preprint arXiv:1802.08232*, 2018.

Tu Bui is currently a Research Fellow at the Centre for Vision Speech and Signal Processing (CVSSP), University of Surrey, United Kingdom. He received BEng and PhD degrees in electrical and electronic engineering at University of Surrey in 2014 and 2019, respectively. His research interests include representation learning, cross-domain retrieval and deep networks.

Daniel Cooper is a Research Software Developer at the Centre for Vision Speech and Signal Processing (CVSSP), University of Surrey. He manages the University of Surrey DLT testbed infrastructure, as well as the design and development of software, integrating AI technologies with DLT.

John Collomosse is full professor at CVSSP, University of Surrey and principal scientist at Creative Intelligence Lab, Adobe Research. His PhD (Bath, 2004) focused on AI for content stylization. He leads a research team focused on the intersection of AI and visual media production, including visual search, performance capture and the fusion of DLT and AI for content provenance.

Mark Bell is a member of the research team The National Archives. Mark started his career working on the AHRC funded 'Traces through Time' project, for probabilistically linking biographical records. His current research interests include machine learning, blockchain, and uncertainty.

Alex Green is the Service Owner for Digital Preservation at The National Archives (UK) and the Product Owner for its digital archive. She is a qualified archivist leading a small research team and is currently interested in new ideas around the contextualization, use and reuse of digital records.

John Sheridan is the Digital Director at The National Archives, with overall responsibility for the organisations digital services and digital archiving capability. His role is to provide strategic direction, developing the people and capability needed for The National Archives to become a disruptive digital archive.

Jez Higgins is an ODI Technical Associate. For the ARCHANGEL project, he worked with CVSSP and TNA colleagues to develop various proof-of-concept web, desktop, and blockchain applications.

Arindra Das is a Senior User Experience Researcher and Product Designer. Arindra has experience working with multinational corporations; government and non-profit organisations; design agencies; and startups. He is also a mentor to numerous early-stage startups and accelerators where he shares his insights on how to design products to solve human problems and create value through effective user research practices. Arindra holds three Masters degree in the topics of computer science, telecommunication, cognitive psychology, design and entrepreneurship including a Dual Masters in Human-Computer Interaction and Design from University College London (UK) & KTH Royal Institute of Technology (Sweden).

Jared Robert Keller is an Emerging Technology Researcher at the ODI. As part of the R&D department, he conducts both qualitative and quantitative research across a wide range of issues related to open data, and supports the ODI's efforts to map complex technological, business, and regulatory landscapes. He specialises in emerging technologies and their implementation in, and impact on, commercial, public sector, and social contexts.

Olivier Thereaux is the head of technology at the Open Data Institute, an independent, non-profit organisation advising governments and organisations worldwide to build a better, more trustworthy data ecosystem. He leads the ODIs work researching emerging tech, prototyping tools and services, and exploring the interplay of technology and policy. Olivier holds a Diplme dIngnieur from Ecole Centrale Paris (now CentraleSupelec).