



(51) International Patent Classification:

G06N 3/096 (2023.01) G06N 3/047 (2023.01)
G06N 3/09 (2023.01) G06N 3/084 (2023.01)
G06N 3/045 (2023.01)

(21) International Application Number:

PCT/EP2024/052855

(22) International Filing Date:

06 February 2024 (06.02.2024)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

20230100091 06 February 2023 (06.02.2023) GR

(71) Applicant: **DEEPMIND TECHNOLOGIES LIMITED**
[GB/GB]; 5 New Street Square, London EC4A 3TW (GB).

(72) Inventors: **DE BALLE PIGEM, Borja**; 6 Pancras Square,
London N1C 4AG (GB). **HAYES, Jamie**; 6 Pancras
Square, London N1C 4AG (GB). **KTENA, Sofia Ira**;

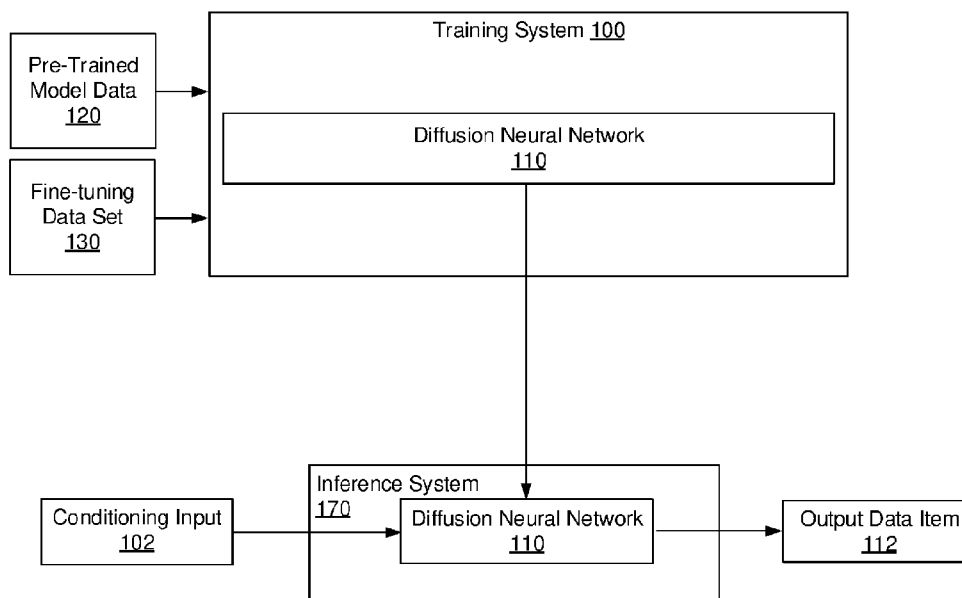
6 Pancras Square, London N1C 4AG (GB). **BERRADA
LANCREY JAVAL, Leonard Alix Jean Eric**; 6 Pan-
cras Square, London N1C 4AG (GB). **WILES, Olivia
Anne**; 6 Pancras Square, London N1C 4AG (GB). **GOW-
AL, Sven Adrian**; 6 Pancras Square, London N1C 4AG
(GB). **SMITH, Samuel Laurence**; 6 Pancras Square, Lon-
don N1C 4AG (GB). **DE, Soham**; 6 Pancras Square, Lon-
don N1C 4AG (GB). **STANFORTH, Robert**; 6 Pancras
Square, London N1C 4AG (GB). **GHALEBIKESABI,
Sahra**; 6 Pancras Square, London N1C 4AG (GB).

(74) Agent: **FISH & RICHARDSON P.C.**; Highlight Business
Towers, Mies-van-der-Rohe-Str. 8, 80807 Munich (DE).

(81) Designated States (unless otherwise indicated, for every
kind of national protection available): AE, AG, AL, AM,
AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ,
CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM,
DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT,
HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG,
KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY,
MA, MD, MG, MK, MN, MU, MW, MX, MY, MZ, NA,

(54) Title: DIFFERENTIALLY PRIVATE DIFFUSION NEURAL NETWORK FINE-TUNING

FIG. 1



(57) **Abstract:** Methods, systems, and apparatus, including computer programs encoded on computer storage media, for differentially private diffusion neural network fine-tuning. In one aspect, a method includes, while training the neural network on a set of fine-tuning data items, for each fine-tuning data item in the set: sampling a set of one or more time steps by sampling from a time step distribution over time steps between a lower bound and an upper bound of the time step distribution, wherein the time step distribution is a non-uniform distribution over the time steps between the lower bound and the upper bound.

NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO,
RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH,
TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS,
ZA, ZM, ZW.

- (84) Designated States** (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— with international search report (Art. 21(3))

DIFFERENTIALLY PRIVATE DIFFUSION NEURAL NETWORK FINE-TUNING

BACKGROUND

This specification relates to generating data items using neural networks.

5 Neural networks are machine learning models that employ one or more layers of nonlinear units to predict an output for a received input. Some neural networks include one or more hidden layers in addition to an output layer. The output of each hidden layer is used as input to or more other layers in the network, i.e., one or more other hidden layers, the output layer, or both. Each layer of the network generates an output from a received input in
10 accordance with current values of a respective set of parameters.

SUMMARY

This specification describes a system implemented as computer programs on one or more computers in one or more locations that trains a diffusion neural network that is used to generate data items (also referred to as “observations”). In particular, the system fine-tunes a
15 pre-trained diffusion neural network through differentially private training.

Particular embodiments of the subject matter described in this specification can be implemented so as to realize one or more of the following advantages.

This specification describes techniques for improving the performance of diffusion neural networks in generating outputs, e.g., in generating images, audio, or video, by fine-
20 tuning a pre-trained diffusion neural network through differentially private training.

Once trained, the diffusion neural network can be used to generate privacy-preserving synthetic versions of the fine-tuning data, e.g., if the fine-tuning data set contains sensitive information that cannot be revealed to others. This can be used, e.g., to train a downstream classifier without giving the system that trains the classifier any access to the fine-tuning data
25 set, or for other suitable applications where access to the fine-tuning data set cannot be granted.

In particular, prior techniques that attempt to generate private synthetic data items are constrained by data availability and cannot generate high-quality synthetic data. Due to their high generation quality, diffusion models are a prime candidate for generating high-quality
30 synthetic data. However, recent studies have found that, by default, the outputs of some diffusion models do not preserve training data privacy.

This specification addresses these issues by privately fine-tuning pre-trained diffusion models to generate useful and, in implementations, provably private synthetic data, even in

applications with significant distribution shift between the pre-training and fine-tuning distributions.

In some particular implementations, by modifying the time step sampling distribution as described below, the system focuses the fine-tuning on the noise regime where preserving privacy is most likely to diminish generation quality, i.e., so that the noise regime where preserving privacy is most likely to harm generation quality is more likely to be sampled during fine-tuning, thereby mitigating this issue and improving final generation quality relative to conventional time step sampling.

Moreover, in some particular implementations the system can account for the small size of the fine-tuning data set by using one or more of (i) sampling multiple time steps per training example and (ii) applying data augmentation to generate multiple noisy data items for each sampled time step.

Thus implementations of the described techniques can facilitate the protection of sensitive information. That is, they enable a neural network, in particular a diffusion neural network, to be trained, i.e., fine-tuned on a set of fine-tuning data items, whilst diminishing the amount of information that can be discovered about the set of fine-tuning data items by accessing the trained diffusion neural network. The trained diffusion neural network can be used for suitable purposes without exposing the fine-tuning data items. As an example, after the fine-tuning the diffusion neural network can be used to generate one or more new data items, based on the fine-tuning data items but without revealing the fine-tuning data items. .

The details of one or more embodiments of the subject matter of this specification are set forth in the accompanying drawings and the description below. Other features, aspects, and advantages of the subject matter will become apparent from the description, the drawings, and the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a diagram of an example training system.

FIG. 2 is a flow diagram of an example process for fine-tuning the diffusion neural network.

FIG. 3 is a flow diagram of an example process for performing a differentially private training step.

FIG. 4 shows an example of the performance of the described techniques.

Like reference numbers and designations in the various drawings indicate like elements.

DETAILED DESCRIPTION

FIG. 1 is a diagram of an example training system 100. The training system 100 is an example of a system implemented as computer programs on one or more computers in one or more locations, in which the systems, components, and techniques described below can be implemented.

This system 100 trains a diffusion neural network 110 that is used to generate data items 112 (also referred to as “observations”).

10 The diffusion neural network 110 is a neural network that, at any given time step, is configured to process a diffusion input that includes (i) a current noisy data item and (ii) data specifying the given time step to generate a diffusion output that defines an estimate of a noise component of the current noisy data item given the current time step.

The estimate of the noise component is an estimate of the noise that has been added to an original data item to generate the current noisy data item.

After training, the system 100 or another inference system 170 uses the trained diffusion neural network 110 to generate an output data item 112 across multiple time steps by performing a reverse diffusion process to gradually de-noise an initial data item until the final output data item 112 is reached. Some or all of the values in the initial data item are noisy values, i.e., are sampled from an appropriate noise distribution.

That is, the initialized data item is the same dimensionality as the final data item but has noisy values. For example, the system 100 can initialize the data item, i.e., can generate the first instance of the data item, by sampling each value in the data item from a corresponding noise distribution, e.g., a Normal distribution or a different noise distribution.

25 That is, the output data item 112 includes multiple values and the initial data item includes the same number of values, with some or all of the values being sampled from a corresponding noise distribution. In some cases, all of the values in the initial data item are sampled from the noise distribution. In some other cases, some of the values are received as input by the system 100 while others are sampled from the noise distribution. For example, 30 the system may be using the diffusion neural network 110 to complete, temporally extend, or otherwise modify an existing data item, and therefore certain values in the initial data item can be taken from the existing data item while the remainder are noisy values.

The diffusion neural network 110 can be configured for use in generating any of a variety of output data items.

For example, the data item can be an image, such that the diffusion neural network 110 is used by the system to perform image generation by generating the intensity values of the pixels of the image.

As another example, the data item can be a video, such that the diffusion neural network 110 is used by the system to perform video generation by generating the intensity values of the pixels of the image frames in the video.

Where the data item comprises an image or video this may have been captured from a real-world environment by a camera or other imaging sensor. For example the image or video may be a medical image or video captured from a patient or from a tissue sample of a patient.

As another example, the data item can be audio data, such that the diffusion neural network 110 is used by the system to generate audio data, e.g., a waveform of audio or a spectrogram, e.g., a mel-spectrogram or a spectrogram where the frequencies are in a different scale, of the audio.

More generally, the data item can be any data item that includes continuous data values. For example, the data item can be an output of a different sensor, e.g., a lidar point cloud, a radar point cloud, an electrocardiogram reading, and so on.

In some examples, the diffusion neural network 110 can be used to perform the reverse diffusion process in a continuous space and then a final discrete data item can be generated from the final data item at the end of the reverse diffusion process. For example, the discrete data item can be generated, e.g., by thresholding or another appropriate discrete diffusion technique. Thus, in these cases, the discrete data item can be, e.g., a sequence of text tokens that each represent a token from a vocabulary of text tokens or an image that has intensity values represented using numerical values with reduced precision.

In some implementations, the diffusion neural network 110 is configured to generate outputs conditioned on an input. For example, the diffusion input at any given time step also can include a conditioning input 102. The conditioning input 102 characterizes one or more properties of a ground truth data item that corresponds to the current noisy data item with the noise component of the noisy data item removed.

For example, the diffusion neural network 110 can be a class-conditional diffusion neural network that generates outputs conditioned on an input representing a specified class of the ground truth data item. For example, when the data items are audio, the conditioning

input can specify a classification for the audio data into a class from a set of possible classes, so that the system generates audio data that belongs to the class. For example, the classes can represent types of musical instruments or other audio emitting devices, i.e., so that the system generates audio that is emitted by the corresponding class, types of animals, i.e., so that the system generates audio that represent noises generated by the corresponding animal, and so on. As another example, when the data items are images or videos, the conditioning input can specify an object class from a plurality of object classes to which an object depicted in the output image or output video should belong.

As another example, the diffusion neural network 110 can be a text-conditional diffusion neural network that generates outputs conditioned on input representing text characterizing one or more properties of the ground truth data item. For example, when the data items are audio, the conditioning input can be text or features of text that the audio should represent, i.e., so that the system serves as a text-to-speech machine learning model that converts text or features of the text to audio data for an utterance of the text being spoken. As another example, the conditioning input can identify a desired speaker for the audio, i.e., so that the system generates audio data that represents speech by the desired speaker.

When the data item is an image or a video, the conditioning input 102 can be a sequence of text and the output data item can be an image or video that describes the text, i.e., the conditioning input can be a caption or other description of the content of the output image or video.

Other types of conditioning are also possible. Some examples of such other types of conditioning when the data items are images now follow.

As yet another particular example, the conditioning input can be an object detection input that specifies one or more bounding boxes and, optionally, a respective type of object that should be depicted in each bounding box.

As yet another particular example, the conditioning input can specify an image at a first resolution and the output data item can comprise the image at a second, higher resolution.

As yet another particular example, the conditioning input can specify an image and the output data item can comprise a de-noised version of the image.

As yet another particular example, the conditioning input can specify an image including a target entity for detection, e.g. a tumor, and the output data item can comprise the

image without the target entity, e.g. to facilitate detection of the target entity by comparing the images.

As yet another particular example, the conditioning input can be a segmentation that assigns each of a plurality of pixels of the output image to a category from a set of categories, e.g., that assigns to each pixel a respective one of the category.

The diffusion neural network 110 can have any appropriate neural network architecture.

For example, the diffusion neural network 110 can be a convolutional neural network, e.g., a U-Net, that has multiple convolutional layer blocks. In some of these cases, the diffusion neural network 110 can include one or more attention layer blocks interspersed among the convolutional layer blocks. As will be described below some or all of the attention blocks can be conditioned on a representation of the conditioning input 102.

As another example, the diffusion neural network 110 can be a Transformer neural network that processes the diffusion input through a set of self-attention layers to generate the denoising output.

The neural network 110 can be conditioned on the conditioning input 102 in any of a variety of ways.

As one example, the system 100 can use an encoder neural network to generate one or more embeddings that represent the conditioning input 102 and the diffusion neural network 110 can include one or more cross-attention layers that each cross-attend into the one or more embeddings.

An embedding, as used in this specification, is an ordered collection of numerical values, e.g., a vector of floating point values or other types of values.

For example, when the conditioning input is text, the system can use a text encoder neural network, e.g., a Transformer neural network, to generate a fixed or variable number of text embeddings that represent the conditioning input.

When the conditioning input is an image, the system can use an image encoder neural network, e.g., a convolutional neural network or a vision Transformer neural network, to generate a set of embeddings that represent the image.

When the conditioning input is audio, the system can use, e.g., an audio encoder neural network, e.g., an audio encoder neural network that has been trained jointly with a decoder neural network as part of a neural audio codec, to generate one or more embeddings that encode the audio.

When the conditioning input is a scalar value, the system can use, e.g., an embedding matrix to map the scalar value or a one-hot representation of the scalar value to an embedding.

Prior to the training, the system 100 obtains pre-trained model data 120 specifying a diffusion neural network 110 that has been trained on a first data set of data items, e.g., obtains data specifying the parameters of the neural network 110 that have been determined as a result of training on the first data set.

That is, the described training that is performed by the system 100 is referred to as “fine-tuning” because the neural network 110 has already been “pre-trained” on another data set prior to being trained by the system 100.

In general, the system 100 can fine-tune any appropriate diffusion neural network 110 that has been trained on an appropriate first data set.

As a particular example, the first data set may be a “public” data set and the diffusion neural network 110 can have been trained on the first data set without employing differentially private training, e.g., using a conventional diffusion model training process, e.g., to optimize a score matching objective.

For example, the score matching objective may be:

$$E_{t,x_0,\epsilon} \|\epsilon - \epsilon_\theta(x_t)\|^2,$$

where E is the expectation operator, ϵ is noise sampled from a noise distribution, x_0 is a training data item sampled from the first data set, t is a time step sampled from a time step distribution over time steps between a lower bound l and an upper bound u of the time step distribution, e.g., from $[0, T]$ or another appropriate interval, and x_t is a noisy data item generated by combining the sampled noise and the sampled training data item in accordance with the sampled time step. For example, for “pre-training,” the time step distribution can be a uniform distribution over the interval.

For example, a given noisy data item x_t can be generated by combining the data item x_0 and the sampled noise ϵ as follows:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon$$

where $\bar{\alpha}_t$ is a noise level that depends on the sampled time step t . In particular, the noise level is a decreasing function of the sampled time step t , so that the larger t is the noisier x_t will be. Examples of such functions include a linear function, a cosine function, and a sigmoid function.

In some cases, the system receives the data 120 specifying the diffusion neural

network 110 after the training has been completed by another system. In some other cases, the system 100 trains the diffusion neural network 110 on the first data set, e.g., without employing differentially private training, e.g., using a conventional diffusion model training process, e.g., to optimize a score matching objective.

5 The system 100 then fine-tunes the trained diffusion neural network on a fine-tuning data set 130 that includes a plurality of fine-tuning data items through differentially private training.

That is, the system 100 uses a modified training scheme that preserves the privacy of the data items in the fine-tuning data set 130 while still training the diffusion neural network
10 110 to generate high-quality data items that appear as if they are drawn from the fine-tuning data set 130.

Performing this “fine-tuning” is described in more detail below.

Once trained, the diffusion neural network 110 can be used, e.g., by the system 100 or the inference system 170, to generate privacy-preserving synthetic versions of the fine-tuning
15 data set 130.

For example, the fine-tuning data set can contain sensitive information that cannot be revealed to others or information that otherwise should remain inaccessible. By using the diffusion neural network 110, the system 170 can generate new, synthetic data items 112 that appear to be drawn from the fine-tuning data set 130, i.e., are realistic data items that can
20 plausibly be drawn from the same distribution as the items in the data set 130, while not revealing any particular one of the data items in the data set 130.

The generated synthetic data items 112 can be used for a variety of purposes. For example, the synthetic data items 112 can be used, e.g., as training data for train a downstream classifier without giving the system that trains the classifier any access to the
25 fine-tuning data set 130.

As another example, the synthetic data items 112 can be used as a validation data set or other data set for testing the performance of a machine learning model.

FIG. 2 is a flow diagram of an example process 200 for fine-tuning a diffusion neural network. For convenience, the process 200 will be described as being performed by a
30 system of one or more computers located in one or more locations. For example, a training system, e.g., the training system 100 depicted in FIG. 1, appropriately programmed in accordance with this specification, can perform the process 200.

The system can repeatedly perform iterations of the process 200 to “fine-tune” (i.e., further train) the diffusion neural network starting from the pre-trained values of the parameters of the diffusion neural network.

The system obtains a set of fine-tuning data items that has been sampled from the fine-tuning data set, e.g., a mini-batch of data items (step 202).

For each fine-tuning data item in the set, the system samples a set of one or more time steps by sampling from a time step distribution over time steps between a lower bound l and an upper bound u of the time step distribution (step 204).

That is, in some implementations, the system samples a single time step per data item while, in other implementations, the system samples multiple times from the distribution for every data item.

Sampling multiple time steps for the same data item can be advantageous when the size of the fine-tuning data set is limited, i.e., because it results in multiple, differently noised data items being generated from a single training data item.

Generally, the time step distribution is a non-uniform distribution over the time steps between the lower bound and the upper bound.

More specifically, during the training on the first data set, i.e., the pre-training data set, the training system can have sampled the time step from a uniform time step distribution over time steps between a lower bound l and an upper bound u of the time step distribution, e.g., uniformly from $[0, T]$ or another appropriate interval, because the model must learn to de-noise images at every noise level.

However, in the fine-tuning scenario, a pre-trained model has already learned that at small time steps the task is to remove small amounts of noise from a natural-seeming data item, and, at large time steps, the task is to project a completely noisy sample closer to the manifold of natural-looking data items. The model behavior at small and large time steps is thus more likely to transfer to different data distributions without further tuning. In contrast, for medium time steps, the model must be aware of the data distribution at hand in order to compose a natural-seeming data item.

Thus, for fine-tuning, the system modifies the training objective so that the time step sampling distribution is not uniform, and instead focuses on training the regimes that contribute more to modelling the key content of a data item, e.g., so that time steps between $[a, b]$ are sampled more frequently than time steps between $[l, a]$ and between $(b, u]$, $a < b < u$.

This non-uniform sampling can help preserve both fine-tuning data item privacy and generated data item quality.

For example, the time step distribution can assign a zero probability to each time step that is greater than or equal to the lower bound and less than or equal to a first threshold value, where the first threshold is less than the upper bound. Thus, in this example, time steps that are less than or equal to the first threshold value are not sampled during fine-tuning even though they were sampled during pre-training.

Instead or in addition, the time step distribution can assign a lower probability to each time step that is less than or equal to the upper bound and greater than or equal to a second threshold value that is less than the upper bound than to any probability that is between the second threshold value and a third threshold value.

For example, the third threshold value can be equal to the first threshold value.

As a particular example, the time step distribution can be a mixture of uniform distributions probability distribution.

In this example, the distribution can satisfy:

$$\sum_i^K w_i U[l_i, u_i],$$

where $\sum_i^K w_i = 1$, $0 \leq l_1, u_K \leq T$ and $u_{k-1} \leq l_k$ for $k \in \{2, \dots, K\}$ and K is the total number of uniform distributions U in the mixture.

For each sampled time step in the set, the system generates one or more new noisy data items by, for each new noisy data item, combining the fine-tuning data item and a respective new noise component for the new noisy data item in accordance with the sampled time step (step 206).

For example, a given noisy data item x_t can be generated by combining the fine-tuning data item x_0 and a respective new noise component ϵ for the new noisy data item as follows:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon$$

where $\bar{\alpha}_t$ is a noise level that depends on the sampled time step t . In particular, the noise level is a decreasing function of the sampled time step t , so that the larger t is the noisier x_t will be.

In some implementations, the system generates a single noisy data item for any given sampled time step.

In some other implementations, multiple different noisy data items are generated for the given sampled time step through the use of data augmentation.

For example, as part of the combining, the system can apply data augmentation to the fine-tuning data item prior to applying noise in accordance with the sampled time step or can generate an initial new data item in accordance with the sampled time step and then apply data augmentation to the initial new data item.

5 The system can apply any appropriate set of augmentations to generate the multiple different noisy data items. For example, when the data items are images, the system can apply one or more of: random crops, random flipping, rotation, and color-jittering.

By leveraging data augmentation, the system can generate a large number of additional noisy data items in order to improve the performance of the fine-tuning process.

10 For each new noisy data item, the system processes a new diffusion input that includes (i) the new noisy data item and (ii) data specifying the sampled time step using the diffusion neural network to generate a new diffusion output that defines an estimate of the new noise component for the sampled time step (step 208).

When the diffusion neural network also receives an input representing a conditioning input, some or all of the fine-tuning data items have a corresponding conditioning input and, when the fine-tuning data item has a corresponding conditioning input, the system includes the input representing the corresponding conditioning input as part of the new diffusion input.

15 The system then trains the diffusion neural network through differentially private training on an objective that measures, for each fine-tuning data item in set, a respective error for each sampled time step and for each new noisy data item (step 210).

Generally, the respective error is an error between the estimate of the respective new noise component for the sampled time step generated by processing the new diffusion input comprising the new noisy data item and the new noise component for the sampled time step.

25 For example, the error can be a score matching objective that is based on a norm of the difference between the estimate of the respective new noise component for the sampled time step generated by processing the new diffusion input comprising the new noisy data item and the new noise component for the sampled time step, e.g., the score matching objective described above.

30 The training is referred to as “differentially private” training, e.g., because the system can modify the gradients of the objective before the gradients are used to update the parameters of the neural network in order to preserve the privacy of the fine-tuning data items in the set. Modifying the gradients can generally include clipping the gradients, adding noise to the gradients, or both.

Generally training a neural network involves backpropagating gradients of the objective. Thus training through differentially private training can involve clipping these gradients, e.g. so that the norm of the gradient does not exceed a threshold, adding noise, e.g. Gaussian noise, or both.

5 The system can generally use any differentially private training algorithm for updating the parameters of the neural network in order to preserve the privacy of the fine-tuning data items in the set. One example of such a technique is described below with reference to FIG. 3.

10 FIG. 3 is a flow diagram of an example process 300 for performing differentially private training. For convenience, the process 300 will be described as being performed by a system of one or more computers located in one or more locations. For example, a training system, e.g., the training system 100 depicted in FIG. 1, appropriately programmed in accordance with this specification, can perform the process 300.

15 In the example of FIG. 3, the set of sampled time steps includes a plurality of time steps, i.e., the system samples multiple times from the non-uniform time step distribution.

The system computes a respective clipped gradient for each fine-tuning data item in the set (step 302).

20 In order to compute the clipped gradient for a given fine-tuning data item, the system computes a respective gradient of the objective for each sampled time step and for each new noisy data item. For example, the system can compute the gradient of the objective for a given sampled time step and a given new noisy data item through backpropagation.

The system then averages the respective gradients for the data item to generate an averaged gradient for the data item and clips the averaged gradient to generate a clipped gradient for the data item.

25 That is, for the i -th averaged gradient $\nabla l_i(w)$, the system computes $\text{clip}_C(\nabla l_i(w))$, where w is a vector of the parameters of the diffusion neural network.

30 “Clipping” an averaged gradient refers to scaling the elements of the averaged gradient so that the averaged gradient has maximal norm of C , i.e., so that the norm of the averaged gradient does not exceed C . Performing this clipping can assist in preserving the privacy of the fine-tuning data items in the set.

For example, the clip_C function can multiply each element of the averaged gradient by the minimum of (i) 1 and (ii) the ratio of C to the norm, e.g., the l2 norm, of the averaged gradient.

The system averages the clipped gradients for the data items in the set to generate an aggregated averaged gradient (step 304).

That is, the system computes $\frac{1}{B} \sum_{i \in B} \text{clip}_C(\nabla l_i(w))$ where B is the total number of clipped gradients.

5 The system applies differentially private noise to the aggregated averaged gradient to generate a noisy gradient (step 306). Here “differentially private noise” is used as a label for any noise that is added in this way; the “differentially private noise” may be, e.g. Gaussian noise.

As a particular example, the system can sample noise ξ from a noise distribution, e.g.,
 10 a normal distribution, and then scales the noise based on a noise variance parameter σ , the total number of clipped gradients B , and the fixed norm C to generate the differentially private noise and then apply the differentially private noise to the aggregated averaged gradient $\frac{1}{B} \sum_{i \in B} \text{clip}_C(\nabla l_i(w))$. In this example, the noisy gradient $\hat{g}$ satisfies:

$$\hat{g} = \frac{1}{B} \sum_{i \in B} \text{clip}_C(\nabla l_i(w)) + \frac{\sigma C}{B} \xi$$

15 The system trains the diffusion neural network using the noisy gradient (step 308). For example, the system can apply an optimizer, e.g., stochastic gradient descent, Adam, AdamW, Adafactor, and so on, to the noisy gradient and the current values of the network parameters of the diffusion neural network to update the current values of the network parameters.

20 By repeatedly performing the process 300, the system can fine-tune the diffusion neural network so that it can be used to generate high-quality data items while preserving the privacy of the individual fine-tuning data items in the fine-tuning data set. That is, because the system uses the noisy gradient in place of the original gradient, the system preserves the privacy of the fine-tuning data items. Because of the way the system samples the time steps
 25 and, optionally, because of the sampling of multiple time steps, the use of data augmentation, or both, the system maintains high generation quality even with the loss of information from the original gradients to the noisy gradients.

FIG. 4 shows an example of the performance of the described techniques. In particular, FIG. 4 shows an example of the performance of a classifier trained on synthetic
 30 data generated using the described techniques (“ours”) and on synthetic data generated using a state of the art (“SOTA”) differential privacy technique for different image generation configurations, i.e., on different data sets, different image resolutions, different diffusion

model sizes, and so on. FIG. 4 also shows an example of “Non-synth” classifiers that are trained on the real data set rather than on synthetic data.

As can be seen from FIG. 4, the classifier achieves improved accuracy when trained on the synthetic images generated using the described techniques, showing that the synthetic data generated by the diffusion neural network after training is high quality (i.e., useful for downstream tasks) while preserving the privacy of the fine-tuning data set used to train the diffusion neural network.

This specification uses the term “configured” in connection with systems and computer program components. For a system of one or more computers to be configured to perform particular operations or actions means that the system has installed on it software, firmware, hardware, or a combination of them that in operation cause the system to perform the operations or actions. For one or more computer programs to be configured to perform particular operations or actions means that the one or more programs include instructions that, when executed by data processing apparatus, cause the apparatus to perform the operations or actions.

Embodiments of the subject matter and the functional operations described in this specification can be implemented in digital electronic circuitry, in tangibly-embodied computer software or firmware, in computer hardware, including the structures disclosed in this specification and their structural equivalents, or in combinations of one or more of them. Embodiments of the subject matter described in this specification can be implemented as one or more computer programs, e.g., one or more modules of computer program instructions encoded on a tangible non transitory storage medium for execution by, or to control the operation of, data processing apparatus. The computer storage medium can be a machine-readable storage device, a machine-readable storage substrate, a random or serial access memory device, or a combination of one or more of them. Alternatively or in addition, the program instructions can be encoded on an artificially generated propagated signal, e.g., a machine-generated electrical, optical, or electromagnetic signal, that is generated to encode information for transmission to suitable receiver apparatus for execution by a data processing apparatus.

The term “data processing apparatus” refers to data processing hardware and encompasses all kinds of apparatus, devices, and machines for processing data, including by way of example a programmable processor, a computer, or multiple processors or computers. The apparatus can also be, or further include, special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application specific integrated circuit). The

apparatus can optionally include, in addition to hardware, code that creates an execution environment for computer programs, e.g., code that constitutes processor firmware, a protocol stack, a database management system, an operating system, or a combination of one or more of them.

5 A computer program, which may also be referred to or described as a program, software, a software application, an app, a module, a software module, a script, or code, can be written in any form of programming language, including compiled or interpreted languages, or declarative or procedural languages; and it can be deployed in any form, including as a stand alone program or as a module, component, subroutine, or other unit
10 suitable for use in a computing environment. A program may, but need not, correspond to a file in a file system. A program can be stored in a portion of a file that holds other programs or data, e.g., one or more scripts stored in a markup language document, in a single file dedicated to the program in question, or in multiple coordinated files, e.g., files that store one or more modules, sub programs, or portions of code. A computer program can be deployed
15 to be executed on one computer or on multiple computers that are located at one site or distributed across multiple sites and interconnected by a data communication network.

 In this specification, the term “database” is used broadly to refer to any collection of data: the data does not need to be structured in any particular way, or structured at all, and it can be stored on storage devices in one or more locations. Thus, for example, the index
20 database can include multiple collections of data, each of which may be organized and accessed differently.

 Similarly, in this specification the term “engine” is used broadly to refer to a software-based system, subsystem, or process that is programmed to perform one or more specific functions. Generally, an engine will be implemented as one or more software
25 modules or components, installed on one or more computers in one or more locations. In some cases, one or more computers will be dedicated to a particular engine; in other cases, multiple engines can be installed and running on the same computer or computers.

 The processes and logic flows described in this specification can be performed by one or more programmable computers executing one or more computer programs to perform
30 functions by operating on input data and generating output. The processes and logic flows can also be performed by special purpose logic circuitry, e.g., an FPGA or an ASIC, or by a combination of special purpose logic circuitry and one or more programmed computers.

 Computers suitable for the execution of a computer program can be based on general or special purpose microprocessors or both, or any other kind of central processing unit.

Generally, a central processing unit will receive instructions and data from a read only memory or a random access memory or both. The essential elements of a computer are a central processing unit for performing or executing instructions and one or more memory devices for storing instructions and data. The central processing unit and the memory can be supplemented by, or incorporated in, special purpose logic circuitry. Generally, a computer will also include, or be operatively coupled to receive data from or transfer data to, or both, one or more mass storage devices for storing data, e.g., magnetic, magneto optical disks, or optical disks. However, a computer need not have such devices. Moreover, a computer can be embedded in another device, e.g., a mobile telephone, a personal digital assistant (PDA), a mobile audio or video player, a game console, a Global Positioning System (GPS) receiver, or a portable storage device, e.g., a universal serial bus (USB) flash drive, to name just a few.

Computer readable media suitable for storing computer program instructions and data include all forms of non volatile memory, media and memory devices, including by way of example semiconductor memory devices, e.g., EPROM, EEPROM, and flash memory devices; magnetic disks, e.g., internal hard disks or removable disks; magneto optical disks; and CD ROM and DVD-ROM disks.

To provide for interaction with a user, embodiments of the subject matter described in this specification can be implemented on a computer having a display device, e.g., a CRT (cathode ray tube) or LCD (liquid crystal display) monitor, for displaying information to the user and a keyboard and a pointing device, e.g., a mouse or a trackball, by which the user can provide input to the computer. Other kinds of devices can be used to provide for interaction with a user as well; for example, feedback provided to the user can be any form of sensory feedback, e.g., visual feedback, auditory feedback, or tactile feedback; and input from the user can be received in any form, including acoustic, speech, or tactile input. In addition, a computer can interact with a user by sending documents to and receiving documents from a device that is used by the user; for example, by sending web pages to a web browser on a user's device in response to requests received from the web browser. Also, a computer can interact with a user by sending text messages or other forms of message to a personal device, e.g., a smartphone that is running a messaging application, and receiving responsive messages from the user in return.

Data processing apparatus for implementing machine learning models can also include, for example, special-purpose hardware accelerator units for processing common and compute-intensive parts of machine learning training or production, e.g., inference, workloads.

Machine learning models can be implemented and deployed using a machine learning framework, .e.g., a TensorFlow framework or a Jax framework.

Embodiments of the subject matter described in this specification can be implemented in a computing system that includes a back end component, e.g., as a data server, or that
5 includes a middleware component, e.g., an application server, or that includes a front end component, e.g., a client computer having a graphical user interface, a web browser, or an app through which a user can interact with an implementation of the subject matter described in this specification, or any combination of one or more such back end, middleware, or front end components. The components of the system can be interconnected by any form or
10 medium of digital data communication, e.g., a communication network. Examples of communication networks include a local area network (LAN) and a wide area network (WAN), e.g., the Internet.

The computing system can include clients and servers. A client and server are generally remote from each other and typically interact through a communication network.
15 The relationship of client and server arises by virtue of computer programs running on the respective computers and having a client-server relationship to each other. In some embodiments, a server transmits data, e.g., an HTML page, to a user device, e.g., for purposes of displaying data to and receiving user input from a user interacting with the device, which acts as a client. Data generated at the user device, e.g., a result of the user
20 interaction, can be received at the server from the device.

While this specification contains many specific implementation details, these should not be construed as limitations on the scope of any invention or on the scope of what may be claimed, but rather as descriptions of features that may be specific to particular embodiments of particular inventions. Certain features that are described in this specification in the context
25 of separate embodiments can also be implemented in combination in a single embodiment. Conversely, various features that are described in the context of a single embodiment can also be implemented in multiple embodiments separately or in any suitable subcombination. Moreover, although features may be described above as acting in certain combinations and even initially be claimed as such, one or more features from a claimed combination can in
30 some cases be excised from the combination, and the claimed combination may be directed to a subcombination or variation of a subcombination.

Similarly, while operations are depicted in the drawings and recited in the claims in a particular order, this should not be understood as requiring that such operations be performed in the particular order shown or in sequential order, or that all illustrated operations be

performed, to achieve desirable results. In certain circumstances, multitasking and parallel processing may be advantageous. Moreover, the separation of various system modules and components in the embodiments described above should not be understood as requiring such separation in all embodiments, and it should be understood that the described program

5 components and systems can generally be integrated together in a single software product or packaged into multiple software products.

Particular embodiments of the subject matter have been described. Other embodiments are within the scope of the following claims. For example, the actions recited in the claims can be performed in a different order and still achieve desirable results. As one

10 example, the processes depicted in the accompanying figures do not necessarily require the particular order shown, or sequential order, to achieve desirable results. In some cases, multitasking and parallel processing may be advantageous.

CLAIMS

1. A method performed by one or more computers, the method comprising:
 - obtaining data specifying a diffusion neural network that has been trained on a first data set of data items and that is configured to process a diffusion input comprising (i) a
 - 5 current noisy data item and (ii) data specifying a current time step to generate a diffusion output that defines an estimate of a noise component of the current noisy data item given the current time step; and
 - fine-tuning the trained diffusion neural network on a fine-tuning data set comprising a plurality of fine-tuning data items through differentially private training, comprising: - 10 obtaining a set of fine-tuning data items from the fine-tuning data set, and for each fine-tuning data item in the set:
 - sampling a set of one or more time steps by sampling from a time step distribution over time steps between a lower bound and an upper bound of the time step distribution, wherein the time step distribution is a non-uniform distribution over the time
 - 15 steps between the lower bound and the upper bound;
 - for each sampled time step in the set:
 - generating one or more new noisy data items by,
 - for each new noisy data item, combining the fine-tuning data item and a respective new noise component for the new noisy data item in accordance with the sampled time step;
 - 20 for each new noisy data item, processing a new diffusion input comprising (i) the new noisy data item and (ii) data specifying the sampled time step using the diffusion neural network to generate a new diffusion output that defines an estimate of the new noise component for the sampled time step; and
 - training the diffusion neural network through differentially private training on an
 - 25 objective that measures, for each fine-tuning data item in the set, a respective error for each sampled time step and for each new noisy data item, the respective error being an error between the estimate of the respective new noise component for the sampled time step generated by processing the new diffusion input comprising the new noisy data item and the new noise component for the sampled time step.
 - 30
2. The method of claim 1, wherein the noisy data item, the fine-tuning data items, and the data items in the first set of data items are images.

3. The method of claim 1, wherein the noisy data item, the fine-tuning data items, and the data items in the first set of data items are audio signals.

4. The method of any preceding claim, wherein the first data set of data items is a public data set and the diffusion neural network has been trained on the first data set of items without employing differentially private training.

5. The method of any preceding claim, wherein the set of sampled time steps comprises a plurality of time steps and wherein training the neural network through differentially private training on the objective comprises:

for each data item in the set:

computing a respective gradient of the objective for each sampled time step and for each new noisy data item;

averaging the respective gradients for the data item to generate an averaged gradient for the data item; and

clipping the averaged gradient to generate a clipped gradient for the data item; and

averaging the clipped gradients for the data items in the set to generate an aggregated averaged gradient;

applying differentially private noise to the aggregated averaged gradient to generate a noisy gradient; and

training the diffusion neural network using the noisy gradient.

6. The method of any preceding claim, wherein generating one or more new noisy data items by, for each new noisy data item, combining the fine-tuning data item and a respective new noise component for the new noisy data item comprises:

sampling the respective new noise component from a noisy distribution;

generating an initial new noisy data item by combining the fine-tuning data item and the respective new noise component in accordance with the sampled time step; and

applying a data augmentation to the initial new noisy data item to generate the new noisy data item.

7. The method of any preceding claim, wherein generating one or more new noisy data items, by for each new noisy data item, combining the fine-tuning data item and a respective new noise component for the new noisy data item comprises, for each new noisy data item:

5 applying a data augmentation to the fine-tuning data item to generate an augmented data item;

 sampling the respective new noise component from a noisy distribution; and

 generating the new noisy data item by combining the augmented data item and the respective new noise component in accordance with the sampled time step.

10

8. The method of any preceding claim, wherein the time step distribution assigns a zero probability to each time step that is greater than or equal to the lower bound and less than or equal to a first threshold value that is less than the upper bound.

15

9. The method of any preceding claim, wherein the time step distribution assigns a lower probability to each time step that is less than or equal to the upper bound and greater than or equal to a second threshold value that is less than the upper bound than to any probability that is between the second threshold value and a third threshold value.

20

10. The method of claim 9, when dependent on claim 8, wherein the third threshold value is equal to the first threshold value.

11. The method of any preceding claim, wherein the time step distribution is a mixture of uniform distributions probability distribution.

25

12. The method of any preceding claim, wherein the diffusion neural network generates output data items conditioned on a conditioning input.

30

13. The method of claim 12, wherein the diffusion neural network is a class-conditional diffusion neural network.

14. The method of claim 12, wherein the diffusion neural network is a text-conditional diffusion neural network.

5

15. The method of any preceding claim wherein the diffusion input further comprises a conditioning input characterizing one or more properties of a ground truth data item that corresponds to the current noisy data item with the noise component of the noisy data item removed.

16. The method of any preceding claim, further comprising:
after fine-tuning the diffusion neural network, using the diffusion neural network to generate one or more new data items.

17. A system comprising:
one or more computers; and
one or more storage devices communicatively coupled to the one or more computers, wherein the one or more storage devices store instructions that, when executed by the one or more computers, cause the one or more computers to perform operations of the respective method of any one of claims 1-16.

18. One or more non-transitory computer storage media storing instructions that when executed by one or more computers cause the one or more computers to perform operations of the respective method of any one of claims 1-16.

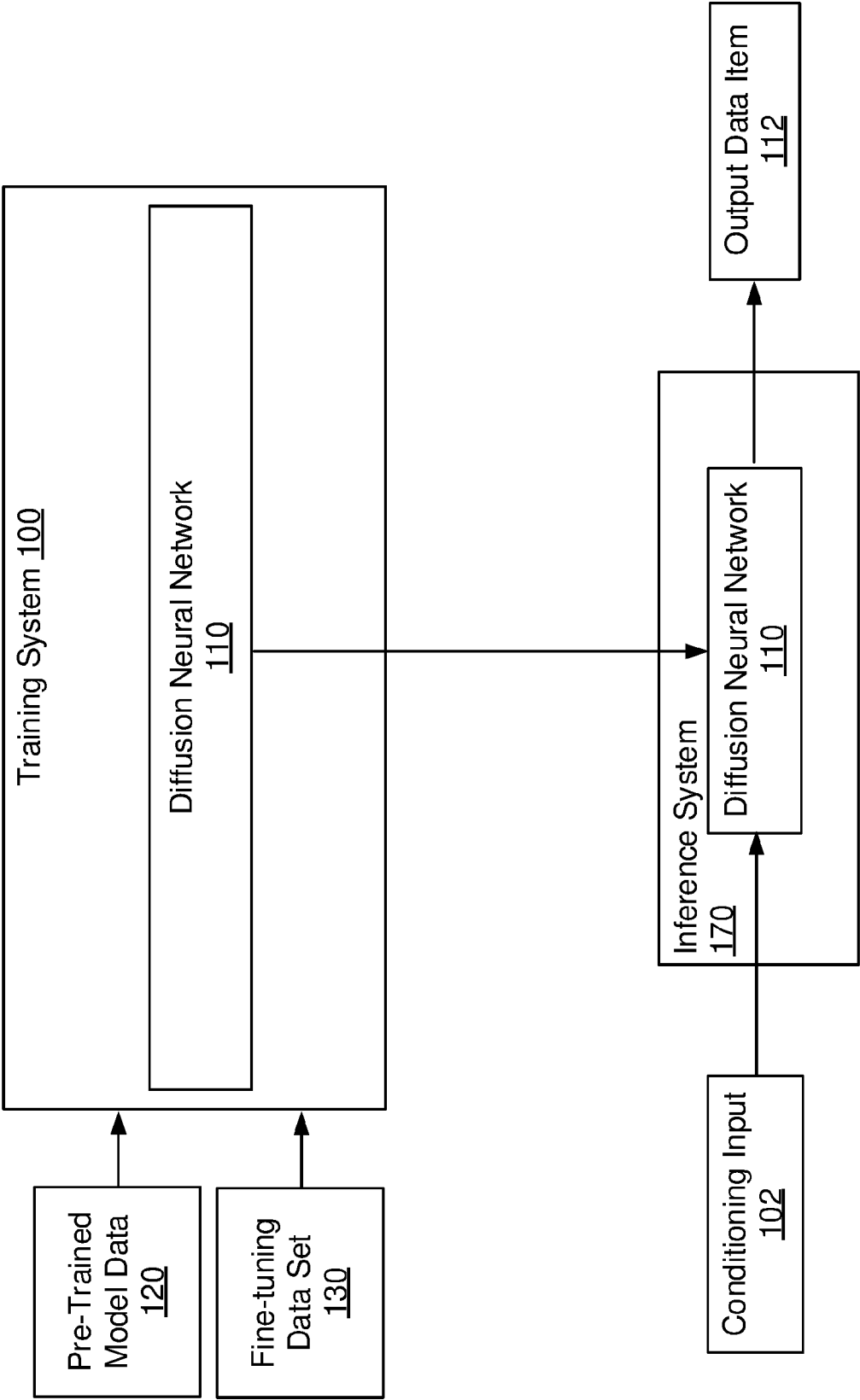


FIG. 1

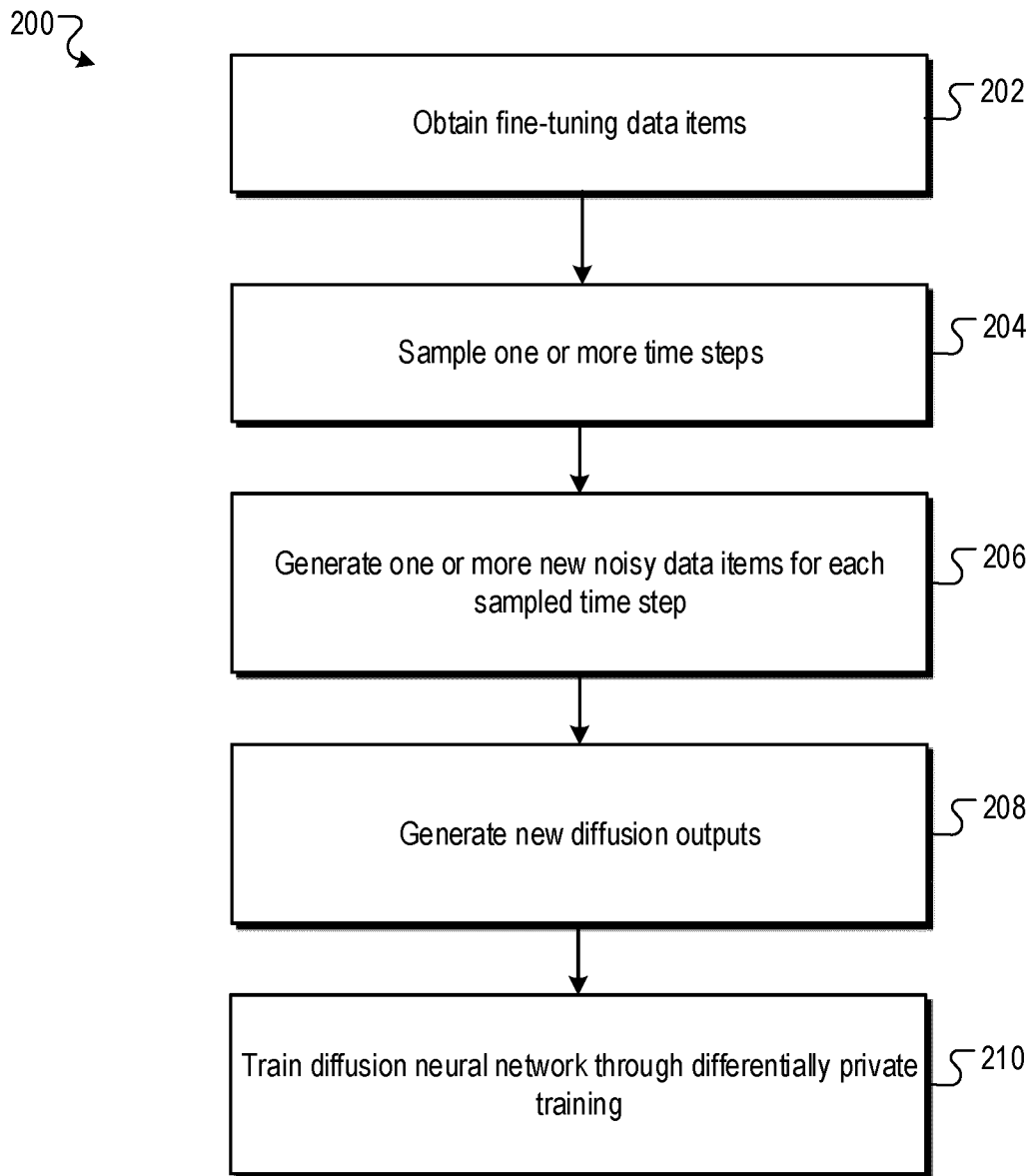


FIG. 2

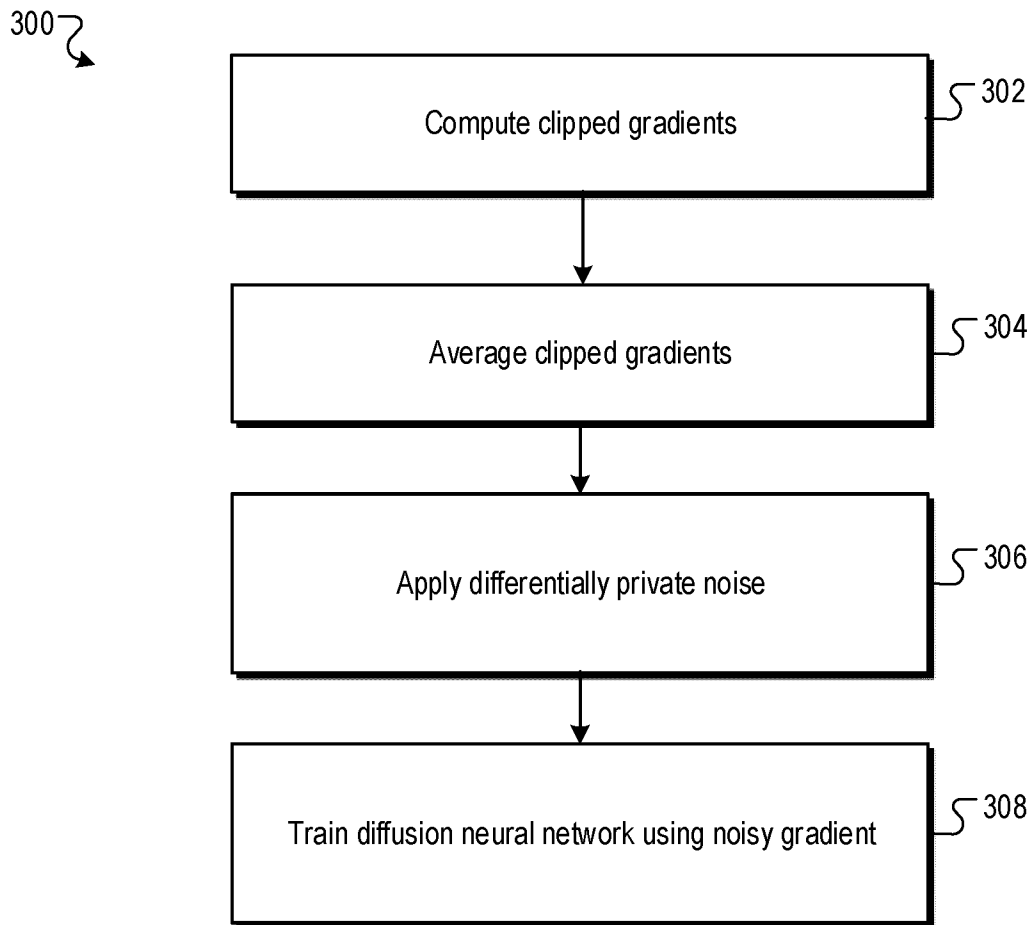


FIG. 3

400 ↗

Dataset	Image Resolution	Diffusion M. Size	Pre-Training Data	Test Accuracy (%)			Prinacy (ϵ, δ)
				SOTA	Ours	Non-synth	
MNIST	28 x 28	4.2M	—	98.1 [Do22]	98.6	99.1	$(10, 10^{-5})$
CIFAR-10	32 x 32 x 3	80.4M	ImageNet32	50.2 [Ha22]	88.0	96.6* [De22]	$(10, 10^{-5})$
Camelyon 17	32 x 32 x 3	80.4M	ImageNet32	—	91.1	90.5	$(10, 3 \cdot 10^{-6})$

FIG. 4

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2024/052855

A. CLASSIFICATION OF SUBJECT MATTER		
INV.	G06N3/096	G06N3/09
	G06N3/045	G06N3/047
	G06N3/084	
ADD.		
According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED		
Minimum documentation searched (classification system followed by classification symbols)		
G06F H04W G06N		
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched		
Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)		
EPO-Internal		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	TIM DOCKHORN ET AL: "Differentially Private Diffusion Models", ARXIV.ORG, CORNELL UNIVERSITY LIBRARY, 201 OLIN LIBRARY CORNELL UNIVERSITY ITHACA, NY 14853, 18 October 2022 (2022-10-18), XP091347618, abstract Sections 1-7; figures 1-4 ----- -/--	1-18
☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.		
* Special categories of cited documents : "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier application or patent but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance;; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance;; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "&" document member of the same patent family		
Date of the actual completion of the international search		Date of mailing of the international search report
1 May 2024		08/05/2024
Name and mailing address of the ISA/ European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016		Authorized officer Malatesta, Lori

INTERNATIONAL SEARCH REPORT

International application No

PCT/EP2024/052855

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	JIA-WEI CHEN: "DPGEN: Differentially Private Generative Energy-Guided Network for Natural Image Synthesis", 2022 IEEE/CVF CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION (CVPR), 1 June 2022 (2022-06-01), pages 8377-8386, XP093155975, DOI: 10.1109/CVPR52688.2022.00820 ISBN: 978-1-6654-6946-3 abstract Sections 4-5; figures 1-3 -----	1-18
T	Sahra Ghalebikesabi: "Differentially Private Diffusion Models Generate Useful Synthetic Images", arXiv.org, 27 February 2023 (2023-02-27), XP093155965, Ithaca DOI: 10.48550/arxiv.2302.13861 Retrieved from the Internet: URL:https://arxiv.org/abs/2302.13861 the whole document -----	1-18