Supplementary Materials

Table 1. Detailed structure and parameters of our DeepGF. Note that w and h are width and height of the feature maps, s is the stride of the convolution operation, c_{in} and c_{out} are the channel numbers of input and output for the building block

<table>
<tr><th colspan="2" rowspan="2">Layers</th><th colspan="2">Feature size ($w \times h \times c$)</th><th colspan="4" rowspan="2">Kernel size, Stride</th></tr>
<tr><th>Attention</th><th>Polar</th></tr>
<tr><td colspan="2">Input</td><td>224x224x3</td><td>224x224x3</td><td colspan="4"></td></tr>
<tr><td colspan="2">Convolution</td><td>112x112x64</td><td>112x112x32</td><td colspan="4">[7x7], 2</td></tr>
<tr><td colspan="2">Max pooling</td><td>56x56x64</td><td>56x56x32</td><td colspan="4">[3x3], 2</td></tr>
<tr><td colspan="2">Building block1</td><td>56x56x64</td><td>56x56x32</td><td colspan="4">[$\cdot$], $s = 1$</td></tr>
<tr><td colspan="2">Building block2</td><td>28x28x128</td><td>28x28x32</td><td colspan="4">[$\cdot$], $s = 2$</td></tr>
<tr><td colspan="2">Building block3</td><td>14x14x128</td><td>14x14x64</td><td colspan="4">[$\cdot$], $s = 2$</td></tr>
<tr><td colspan="2">Building block4</td><td>7x7x256</td><td>7x7x64</td><td colspan="4">[$\cdot$], $s = 2$</td></tr>
<tr><td colspan="2">Concatenation</td><td colspan="2">7x7x320</td><td colspan="4"></td></tr>
<tr><td colspan="2">Global average pooling</td><td colspan="2">1x1x320</td><td colspan="4">[7x7]</td></tr>
<tr><td colspan="2">VTI-LSTM1 & VTI-LSTM2</td><td colspan="2">320 & 320</td><td colspan="4"></td></tr>
<tr><td colspan="2">FC1 & FC2</td><td colspan="2">256 & 2</td><td colspan="4"></td></tr>
<tr><td rowspan="4">Building block</td><td rowspan="2">Convolution</td><td colspan="2">$\frac{w}{s} \times \frac{h}{s} \times c_{\text{in}}$</td><td rowspan="2">[3x3], s</td><td>[3x3], s</td><td>[5x1], s</td><td>[7x1], s</td></tr>
<tr><td colspan="2">$\frac{w}{s} \times \frac{h}{s} \times c_{\text{in}}$</td><td>[3x3], 1</td><td>[1x5], 1</td><td>[1x7], 1</td></tr>
<tr><td>Concatenation</td><td colspan="2">$\frac{w}{s} \times \frac{h}{s} \times 2c_{\text{in}}$</td><td colspan="4"></td></tr>
<tr><td>Convolution</td><td colspan="2">$\frac{w}{s} \times \frac{h}{s} \times c_{\text{out}}$</td><td colspan="4">[1x1], 1</td></tr>
</table>

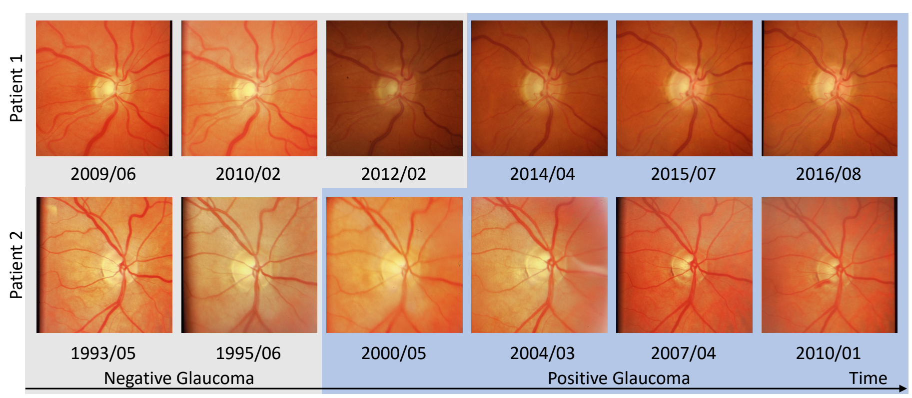

Fig. 1. Sequential fundus images in the SIGF database.

Algorithm 1: Details of the AC training strategy.

Input: The training DeepGF network; the training set of the fundus image sequences $\{\mathbf{I}^s\}_{s=1}^{S}$ and their corresponding glaucoma labels $\{l_{t+1}^s\}_{t=1,s=1}^{T,S}$, where $\mathbf{I}^s$ is the training sequence ($\mathbf{I}^s = \{\mathbf{I}_t^s\}_{t=1}^{T}$) and S is the number of the training sequences.

Output: Learned parameters ω of the DeepGF.

Initialize: Maximum number of the training epoches N; the batch size number B; the training iteration $j = 1$; the learning rate α; the discarding rate β; the discarding epoch interval δ; the maximum discarding times M; the discarding time $m = 1$; the threshold of the loss gap $\mathrm{th_g}$; the threshold of convergence loss $\mathrm{th_c}$.

Run the DeepGF with the initial parameters ω^0.

while $n < N$ **do**
- $n \leftarrow n{+}1$.
- **for** $i = 1 \rightarrow \frac{S}{B}$ **do**
  - Randomly sample B image sequences $\{\mathbf{I}^b\}_{b=1}^{B}$ from $\{\mathbf{I}^s\}_{s=1}^{S}$.
  - Calculate $\mathcal{L}_f(\omega^j)$ with the input of $\{\mathbf{I}^b\}_{b=1}^{B}$ according to (**??**).
  - Calculate $\nabla_\omega \mathcal{L}_f(\omega)$ by $\nabla_\omega \mathcal{L}_f(\omega) = -\frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{1+e^{-z_t}} - l_{t+1}\right) \cdot \nabla_\omega z(\omega)$.
  - $\omega^{j+1} \leftarrow \omega^j - \alpha \cdot \nabla_\omega \mathcal{L}_f(\omega)$.
  - $j \leftarrow j{+}1$.
- **end**
- **if** $\mathcal{L}_f(\omega^j) < \mathrm{th_c}$ **then**
  - **Break from the while loop**.
- **end**
- **else if** $|\mathcal{L}_f(\omega^j|l_{t+1}=0) - \mathcal{L}_f(\omega^j|l_{t+1}=1)| < \mathrm{th_g}$ **then**
  - **Break from the current loop**.
- **end**
- **else if** $\mathrm{mod}(n, \delta) = 0$ **and** $m \leq M$ **then**
  - Calculate the forecast loss across the whole training set: $\{\mathcal{L}_f^s(\omega^j)\}_{s=1}^{S}$.
  - Sort $\{\mathbf{I}^s\}_{s=1}^{S}$ according to the value of $\{\mathcal{L}_f^s(\omega^j)\}_{s=1}^{S}$ in descending order.
  - Update the training set to $\{\mathbf{I}^s\}_{s=1}^{S-\beta S}$ by discarding the last βS sequences from $\{\mathbf{I}^s\}_{s=1}^{S}$, which rank the lowest in terms of $\mathcal{L}_f(\omega^j)$.
  - $S \leftarrow S - \beta S$.
  - $m \leftarrow m + 1$.
- **end**

end

return Trained parameters ω^j.

Table 2. Implementation details of training our DeepGF. Note that the hyper-parameters of the AP-CNN and VTI-LSTM are tuned to minimize the divergence between forecast and groundtruth labels over the validation set.

p and q in the VTI gate	-0.4 and 0.5
$\triangle y_t$ at the first time step in the VTI gate	0
Time step T of the VTI-LSTM	5
Learning rate α for Adam optimization	1×10^{-4}
Exponential decay rate β_1 and β_2 for Adam optimization	0.9 and 0.99
Epsilon ϵ for Adam optimization	1×10^{-8}
Training epoches N	300
Batch size	4
Classification threshold ξ	0.5
Discarding rate β	0.5
Discarding epoch interval	3
Total discarding times M	6
GPU	Nvidia GTX 1080 Ti
Forecast speed	107 image sequences per second