1. Overview

This supplementary material is organized as follows:

- Sec. 2 presents additional **comparison results**.
- Sec. 3 provides more detailed **ablation studies**.
- Sec. 4 provides the experimental results from **human evaluation**.
- Sec. 5 introduces preliminary concepts on **diffusion models**.
- Sec. 6 reviews preliminaries on **graph convolutional networks**.
- Sec. 7 summarizes preliminary knowledge on **Connectionist Temporal Classification**.
- Sec. 8 addresses several **potential questions**.
- Sec. 9 presents additional **architectural details** of the proposed model.
- Sec. 10 outlines the training and inference procedures of FGDM in **algorithmic** form.
- Sec. 11 gives detailed descriptions of the **evaluation metrics**.
- Sec. 12 provides additional **data processing** details.
- Sec. 13 includes more **visualization results**.
- Sec. 14 discusses **limitations and future works**.

Table 6. Comparison of ASGCN with existing GCN variants in the FGDM architecture on PHOENIX14T.

Methods	DEV			TEST		
	B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
Baseline	22.54	7.39	81.00	22.41	7.77	79.76
FGDM (ST-GCN [20])	24.04	8.66	77.77	24.17	8.82	75.28
FGDM (AGCN [14])	24.99	9.02	77.64	24.20	8.85	74.90
FGDM (AAGCN [15])	25.76	8.69	77.37	23.96	8.60	75.02
FGDM (ASGCN)	**28.09**	**9.84**	**76.57**	**26.29**	**9.25**	**73.91**

2. Additional Comparisons

In this section, we present more fine-grained comparison experiments to thoroughly validate the effectiveness, transferability, and real-time capability of the proposed method.

Comparing ASGCN with existing GCN variants within the FGDM architecture. As shown in Tab. 6, we evaluate several GCN variants, including ST-GCN [20], AGCN [14], and AAGCN [15], within the unified FGDM architecture. Compared with the Baseline, all FGDM-based models achieve substantial improvements. For example, FGDM (ST-GCN) increases BLEU-1 by +1.50%/+1.76% and BLEU-4 by +1.27%/+1.05%, while reducing WER by -3.23%/-4.48%. These improvements result from the two-stage design of FGDM, which simultaneously captures fine-grained joint-level dependencies and global semantic coherence. In contrast, the Baseline struggles to model such detailed dependencies and therefore shows lower performance. Notably, the proposed ASGCN consistently outperforms all other variants across all metrics. It achieves a BLEU-1 score of 28.09% on DEV, surpassing the strongest competitor AAGCN by +2.33%. BLEU-4 improves by +0.84%/+0.40%, and WER decreases by -0.99% on TEST. This performance advantage is attributed to ASGCN's ability to preserve skeletal topology while adaptively adjusting joint dependency strengths according to semantic and contextual correlations, which makes it more suitable for modeling the complex pose patterns in sign language. In summary, two important conclusions can be drawn: 1) All FGDM-based architectures significantly outperform the Baseline, confirming the effectiveness of the FGDM design; 2) ASGCN achieves the best overall performance among existing GCN variants, demonstrating its strong capability in sign language pose modeling.

Table 7. Comparison between SCG Loss and CTC Loss on PHOENIX14T.

Methods	DEV			TEST		
	B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
Baseline	22.54	7.39	81.00	22.41	7.77	79.76
+SCG (w/ CTC Loss)	20.75	6.26	89.08	20.47	6.56	88.73
+SCG (w/ SCG Loss)	**25.13**	**9.12**	**78.92**	**24.26**	**9.17**	**78.12**

Comparison between SCG loss and CTC loss. The SCG Loss is an improved version of the CTC loss, specifically designed for supervising diffusion models. As shown in Tab. 7, we compare the performance of SCG Loss and CTC Loss. When using the original CTC Loss, the baseline performance drops significantly: BLEU-1 decreases by -1.79% (DEV) and -1.13% (TEST), BLEU-4 decreases by -1.13%/-1.21%, and WER increases by +8.08%/+8.97%. In contrast, when using SCG Loss, all metrics improve significantly: BLEU-1 increases by +2.59%/+1.85%, BLEU-4 increases by +1.73%/+1.40%, and WER decreases by -2.08%/-1.64%. The large performance gap between the two losses arises from their ability to perceive diffusion stages. CTC Loss is unaware of the model's diffusion stage and thus applies uniform supervision to both early noisy representations and later high-level features, which hinders convergence and weakens semantic guidance. In contrast, SCG Loss is explicitly designed to perceive the diffusion stage. It allocates greater attention to semantically rich features in the later stages while suppressing the influence of early-stage noise. This stage-aware mechanism yields more stable convergence and stronger semantic alignment, ultimately enhancing the overall generation quality. In summary, this comparison highlights the clear advantage of SCG Loss in guiding diffusion models toward semantically consistent and higher-quality pose generation.

Exploring the transferability of SCG. To explore the transferability of SCG, we conduct experiments on different diffusion models. As shown in Tab. 8, all diffusion models achieve improved WER after applying SCG. Notably, the largest WER reductions occur in FGDM (ST-

Table 8. Effect of SCG across different diffusion models on PHOENIX14T.

Methods	DEV			TEST		
	B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
Baseline	22.54	7.39	81.00	22.41	7.77	79.76
w/ SCG	**25.13**	**9.12**	**78.92**	**24.26**	**9.17**	**78.12**
Sign-IDD [17]	25.13	8.87	79.08	23.16	8.22	79.15
w/ SCG	**25.45**	**8.97**	**78.31**	**25.18**	**9.34**	**76.24**
FGDM (ST-GCN [20])	24.04	8.66	77.77	24.17	8.82	75.28
w/ SCG	**24.74**	**9.08**	**73.42**	**24.17**	**9.17**	**71.28**
FGDM (AGCN [14])	24.99	**9.02**	77.64	24.20	8.85	74.90
w/ SCG	**25.10**	8.61	**76.65**	**24.25**	**8.94**	**73.44**
FGDM (AAGCN [15])	**25.76**	8.69	77.37	**23.96**	8.60	75.02
w/ SCG	23.66	**8.88**	**73.93**	23.86	**8.78**	**72.34**
FGDM (ASGCN)	**28.09**	**9.84**	76.57	26.29	9.25	73.91
w/ SCG	26.92	9.48	**72.22**	**26.51**	**9.67**	**70.70**

Table 9. Comparison of FLOPs, parameters, and inference time (for generating a 200-frame sign poses).

Method	FLOPs (G)	Params (M)	Time (s)
PT-GN [12]	38.930	14.870	0.46
Sign-IDD [17]	5.525	12.479	0.085
Baseline	7.989	16.285	0.067
FGDM (ours)	9.075	20.397	0.105

GCN) and FGDM (ASGCN), reaching -4.35% /-4.00% and -4.35%/-3.21%, respectively. SCG's consistent WER reduction stems from its mechanism to ensure robust semantic consistency between the generated poses and glosses, which directly translates into more accurate gloss decoding by the back-translation model. In addition, BLEU-1 and BLEU-4 scores consistently improve for Baseline, Sign-IDD, and FGDM (ST-GCN). Although other models do not show consistent BLEU improvements, their scores do not decline overall, and the substantial, consistent improvement in WER serves as compelling evidence that SCG successfully enhances overall model performance. In conclusion, the experimental results demonstrate that SCG successfully transfers across diverse diffusion models, consistently delivering performance improvements, particularly in the crucial metric of WER.

Comparison of FLOPs, parameters, and inference time. We compare the FLOPs, parameters, and inference time of FGDM with existing SLP methods, including PT-GN, Sign-IDD, and the baseline. As shown in Tab. 9, all methods generate 200-frame sign pose sequences and are evaluated 100 times to obtain the averaged results. FGDM contains 20.37 M parameters, 9.075 GFLOPs, and achieves an inference time of 0.105 s. Compared with the baseline, this corresponds to an increase of +1.176 GFLOPs, +4.139 M parameters, and +0.038 s inference time. Although FGDM introduces moderate computational overhead, it still maintains real-time performance and achieves substantial performance gains (see Tab. 1 and Tab. 2). Future work may focus on further reducing the inference cost of FGDM.

Table 10. Ablations for the order of the Focal and General stages in FGDM on PHOENIX14T.

Methods	DEV			TEST		
	B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
Baseline	22.54	7.39	81.00	22.41	7.77	79.76
General → Focal	24.05	8.15	82.31	23.25	8.63	81.45
Focal → General	**28.09**	**9.84**	**76.57**	**26.29**	**9.25**	**73.91**

Table 11. Ablations for SCG with and without the Semantic Decoder (w/ SD and w/o SD) on PHOENIX14T.

Methods	DEV			TEST		
	B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
Baseline	22.54	7.39	81.00	22.41	7.77	79.76
+SCG (w/o SD)	7.41	0.34	100.00	6.53	0.00	100.00
+SCG (w/ SD)	**25.13**	**9.12**	**78.92**	**24.26**	**9.17**	**78.12**

3. Additional Ablations

In this section, we conduct extensive ablation studies on various hyperparameters and architectural designs to thoroughly validate the rationality of the proposed method.

Ablations for the order of the Focal and General stages. As shown in Tab. 10, the Focal→General configuration significantly outperforms the General→Focal variant across all metrics, with WER reduced by –5.40% / –7.54%, whereas the General→Focal order even results in higher WER than the baseline (+1.31% / +1.69%). This degradation occurs because, in the General→Focal order, the Focal stage cannot effectively model joint-level dependencies. The General stage first embeds all joint information into a high-dimensional vector of size d_g, while the Focal stage expects joint-structured features of shape $J \times d_f$. Converting from General to Focal therefore requires reconstructing joint-level representations ($d_g \rightarrow J \times d_f$). However, after the General stage, distinctions among individual joints become blurred, hindering the graph convolution operations in the Focal stage. In contrast, applying the Focal stage before the General stage avoids this issue, enabling the model to fully leverage both local joint-level modeling and global sequence modeling. Therefore, FGDM adopts the Focal→General design as its final configuration.

Ablations for Semantic Decoder. The Semantic Decoder aims to bridge the semantic gap introduced by cross-modal semantic guidance, a crucial aspect that previous works have largely overlooked [7, 16]. As shown in Tab. 11, removing the Semantic Decoder leads to a significant performance drop when using SCG—the WER on both DEV and TEST rises to 100%, even largely worse than the base-

Table 12. Ablations for SCG with and without the V2S Adapter (w/ Adapter and w/o Adapter) on PHOENIX14T.

Methods	DEV			TEST		
	B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
Baseline	22.54	7.39	81.00	22.41	7.77	79.76
+SCG (w/o Adapter)	21.59	7.29	82.87	21.53	7.35	82.08
+SCG (w/ Adapter)	**25.13**	**9.12**	**78.92**	**24.26**	**9.17**	**78.12**

Table 13. Ablations for LSD and GSD on PHOENIX14T.

Methods	DEV			TEST		
	B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
-	**25.13**	**9.12**	**78.92**	**24.26**	**9.17**	**78.12**
w/o LSD	20.17	6.40	89.62	19.50	6.19	89.36
w/o GSD	20.85	6.94	86.39	20.81	7.11	85.98

line. This is because the large discrepancy between visual and linguistic modalities makes effective alignment impossible, and the misaligned loss instead misguides the training. In contrast, incorporating the Semantic Decoder greatly improves performance compared to the baseline, reducing WER to 78.92 % (DEV) and 78.12 % (TEST). This demonstrates that the Semantic Decoder bridges the modality gap and allows the alignment loss to function as intended.

Ablations for V2S Adapter. The V2S Adapter, placed before the Semantic Decoder, bridges visual features to prevent degradation from direct decoding. As shown in Tab. 12, removing it (w/o Adapter) leads to performance drops compared with the baseline: BLEU-1 decreases by –0.95% (DEV) / –0.88% (TEST), BLEU-4 by –0.10% / –0.42%, and WER increases by +1.87% / +2.32%. In contrast, using the V2S Adapter (w/ Adapter) allows SCG to consistently improve baseline performance. These results demonstrate that the Semantic Decoder struggles to directly decode raw visual features, while the V2S Adapter serves as an essential intermediate module that refines and transforms visual representations for better decoding. Overall, this ablation validates the rationality and necessity of the V2S Adapter.

Ablations for LSD and GSD. LSD and GSD are the main components of the Semantic Decoder, responsible for decoding local and global semantics, respectively. As shown in Tab. 13, removing either LSD or GSD significantly degrades performance, as insufficient semantic decoding directly weakens the subsequent SCG alignment. In particular, removing LSD causes the most severe performance drop (BLEU-1: –4.96%/–4.76%, BLEU-4: –2.72%/–2.89%, WER: +10.70%/+11.24%), since LSD also performs downsampling. Without it, the overlong feature sequence misaligns with the shorter gloss sequence, leading to spiking and numerical instability that severely harm performance [3, 8, 10, 22]. In summary, this ablation validates the necessity of both LSD and GSD for sufficient semantic decoding and stable alignment in SCG.

Table 14. Ablations for the internal structure of LSD on PHOENIX14T. Kn and Pn denote a 1D convolutional layer and a pooling layer with a kernel size of n, respectively.

Methods	DEV			TEST		
	B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
{K5,K5}	21.64	6.64	85.00	22.53	7.29	84.29
{K5,P2}	24.71	8.34	79.07	22.72	8.07	79.55
{K5,P2,K5}	**25.13**	**9.12**	**78.92**	**24.26**	**9.17**	**78.12**
{K5,P2,K5,P2}	23.32	8.24	79.93	22.72	7.89	78.07

Table 15. Ablations for the number of BiLSTM layers η in GSD and the hyperparameter α in Eq. 10 on PHOENIX14T.

Methods		DEV			TEST		
		B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
η	1	25.13	**9.12**	**78.92**	**24.26**	**9.17**	78.12
	2	24.14	8.26	79.77	23.11	8.23	**78.05**
	3	**25.61**	9.01	80.28	23.61	8.63	78.82
α	1	22.74	7.77	**78.49**	22.56	7.90	**77.13**
	10	**25.13**	**9.12**	78.92	**24.26**	**9.17**	78.12
	100	23.25	7.96	78.78	22.59	7.94	78.68

Ablations for the internal structure of LSD. To evaluate the design of LSD, we conduct ablation experiments on its internal components in Tab. 14. According to the results, the configuration {K5, P2, K5} achieves the best overall performance across all metrics, with BLEU-1 reaching 25.13%/24.26% and BLEU-4 reaching 9.12%/9.17%, significantly outperforming other settings. The pooling layer P2 plays a crucial role, as it performs downsampling to prevent excessively long output sequences, which can otherwise impair alignment with gloss sequences. In contrast, the configuration {K5, K5}, which lacks a pooling layer, yields notably poorer results (BLEU-1: 21.64%/22.53%, BLEU-4: 6.64%/7.29%). However, adding too many pooling layers also shortens the sequence excessively, leading to degraded performance—as seen in {K5, P2, K5, P2}, where BLEU-1 drops by -1.81%/-1.54% and BLEU-4 by -0.88%/-1.28% compared to {K5, P2, K5}. In summary, we adopt {K5, P2, K5} as the final configuration for the LSD.

Ablations on the number of BiLSTM layers (η) in GSD and the α in Eq. 10. As shown in Tab. 19, we first ablate the number of BiLSTM layers η in the GSD module. Increasing η from 1 to 3 slightly improves some metrics (e.g., $\eta = 2$ yields a -0.07% WER improvement on TEST, and $\eta = 3$ increases BLEU-1 by +0.48% on DEV), but overall performance declines on most other metrics. Therefore, $\eta = 1$ is identified as the optimal configuration. For α in Eq. 10, which controls the sensitivity of SCG loss to the diffusion stage (larger values indicate higher sensitivity), we evaluate $\alpha \in 1, 10, 100$. When α increases

Table 16. Ablations for the number of sampling steps I on the PHOENIX14T.

Methods	DEV			TEST		
	B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
1	26.11	9.05	74.03	25.38	9.42	72.55
5	**26.92**	**9.48**	**72.22**	**26.51**	**9.67**	**70.70**
10	26.64	9.13	72.51	26.27	9.30	71.43

Table 17. Ablations for the TCN module in the Focal stage and its kernel size on PHOENIX14T. Kn denotes a TCN layer with a kernel size of n.

Methods	DEV			TEST		
	B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
w/o TCN	23.90	8.43	78.38	23.86	8.49	77.11
w/ TCN (K7)	27.09	9.59	77.18	25.49	**9.53**	75.70
w/ TCN (K9)	**28.09**	**9.84**	76.57	**26.29**	9.25	**73.91**
w/ TCN (K11)	25.67	9.02	**75.85**	24.48	8.74	75.23

Table 18. Ablations for the computation method of Eq. 3 on PHOENIX14T.

Methods	DEV			TEST		
	B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
$\mathbf{A}_a^i + \mathbf{A}_b^i + \mathcal{M}^i$	26.87	9.37	78.17	25.21	9.21	76.17
$\mathbf{A}_a^i \odot \mathbf{A}_b^i \odot \mathcal{M}^i$	26.61	9.26	**75.79**	26.23	**9.74**	74.08
$(\mathbf{A}_a^i + \mathbf{A}_b^i) \odot \mathcal{M}^i$	**28.09**	**9.84**	76.57	**26.29**	9.25	**73.91**

from 1 to 10, BLEU-1 improves by +2.39%/+1.00% and BLEU-4 by +1.35%/+1.27%, while WER slightly worsens (+0.43%/+0.99%), resulting in an overall performance gain. However, further increasing α to 100 degrades all metrics (BLEU-1: -1.88%/-1.67%, BLEU-4: -1.16%/-1.23%, WER: +0.14%/+0.56%). In summary, we adopt $\eta = 1$ and $\alpha = 10$ as the final model configuration.

Ablations for the sampling steps I. As shown in Fig. 16, using $I = 5$ sampling steps during inference yields consistently better performance than $I = 1$ or $I = 10$. Therefore, we adopt $I = 5$ as our final configuration.

Ablations for the TCN in the Focal stage and its kernel size. The TCN in the Focal stage compensates for ASGCN's limitation in explicitly modeling temporal dependencies. As shown in Tab. 17, incorporating the TCN significantly improves performance. Specifically, w/ TCN (K7) achieves a +3.19%/+1.63% improvement in BLEU-1, a +1.16%/+1.04% improvement in BLEU-4, and a WER reduction of -1.20%/-1.41% compared to w/o TCN. We further explore different kernel sizes (7, 9, and 11) and find that a kernel size of 9 yields the best overall performance (BLEU-1: 28.09%/26.29%, BLEU-4: 9.84%/9.25%, WER: 76.50%/73.91%). These results clearly demonstrate the importance of the TCN in the Focal stage, and we thus adopt a kernel size of 9 in our final configuration.

Ablations for the computation method of Eq. 3. As shown in Tab. 18, we compare three formulations for computing the adjacency matrix $\mathbf{A}^i$ in Eq. 3: $\mathbf{A}_a^i + \mathbf{A}_b^i + \mathcal{M}^i$, $\mathbf{A}_a^i \odot \mathbf{A}_b^i \odot \mathcal{M}^i$, and $(\mathbf{A}_a^i + \mathbf{A}_b^i) \odot \mathcal{M}^i$. The results indicate that $(\mathbf{A}_a^i + \mathbf{A}_b^i) \odot \mathcal{M}^i$ consistently outperforms the additive form $\mathbf{A}_a^i + \mathbf{A}_b^i + \mathcal{M}^i$, achieving +1.22%/+1.08% improvements in BLEU-1 (DEV/TEST), +0.47%/+0.04% in BLEU-4, and -1.60%/-2.26% reductions in WER. This is because the pure additive form struggles to balance the three information sources, leading to unstable optimization and weakened representation learning. In comparison, although the pure multiplicative form $\mathbf{A}_a^i \odot \mathbf{A}_b^i \odot \mathcal{M}^i$ performs slightly better on WER (+0.78% DEV) and BLEU-4 (-0.49% TEST), its overall performance is inferior. This is mainly due to excessive information loss — the element-wise product only retains edges where all three matrices have simultaneously high responses, causing most connections to vanish and gradients to propagate sparsely. In contrast, $(\mathbf{A}_a^i + \mathbf{A}_b^i) \odot \mathcal{M}^i$ achieves the best balance. The gated formulation allows $\mathcal{M}^i$ to act as a semantic modulator, adaptively filtering and refining the fusion of topological and correlation-based structures while maintaining stable gradient flow. Therefore, we adopt $(\mathbf{A}_a^i + \mathbf{A}_b^i) \odot \mathcal{M}^i$ as the final formulation.

Table 19. Ablations for the number of neighbors n in Eq. 4 and the loss weight γ in Eq. 18 on PHOENIX14T.

Methods		DEV			TEST		
		B1↑	B4↑	WER↓	B1↑	B4↑	WER↓
n	1	25.77	9.06	77.38	24.97	8.90	75.78
	2	26.11	9.20	**75.63**	25.04	8.97	75.60
	3	**28.09**	9.84	76.57	26.29	9.25	**73.91**
	4	27.23	**10.33**	76.70	**26.42**	**9.47**	74.08
γ	0.001	17.45	4.93	93.62	1.01	5.39	93.59
	0.0001	24.14	8.26	79.77	23.11	8.23	**78.05**
	0.00001	**25.13**	**9.12**	**78.92**	**24.26**	**9.17**	78.12
	0.000001	23.06	8.29	83.19	22.71	8.25	81.80

Ablations for parameter n in Eq. 4 and γ in Eq. 18. As shown in Tab. 19, we perform ablations on the hyperparameters n and γ. For the number of neighbors n, when it increases from 1 to 2, all metrics on both DEV and TEST sets improve. Increasing n from 2 to 3 yields further gains, except for a slight WER rise of +0.94% (DEV); notably, BLEU-1 improves by +1.98% (DEV) and WER drops by -1.69% (TEST). However, increasing n from 3 to 4 results in a BLEU-1 decline of -0.86% (DEV) and WER increases of +0.13%/+0.17% (DEV/TEST), with less noticeable improvements elsewhere. For the SCG loss weight γ, reducing it from 0.001 to 0.0001 significantly boosts performance, particularly with WER reductions of -13.85%/-15.54%. Further lowering γ to 0.00001 continues to improve most metrics except for a minor WER increase of +0.07% (TEST). However, an excessively small value (0.000001) degrades performance. Based on the above ob-

Table 20. Human evaluation voting results, including per-sample votes for each model, total votes, and overall vote percentages.

Sequence_ID	Votes		
	PT [12]	Sign-IDD [17]	FGDM (ours)
19October_2009_Monday_tagesschau-229	0	2	8
19February_2010_Friday_tagesschau-4101	0	3	7
09August_2009_Sunday_tagesschau-1180	0	1	9
05December_2009_Saturday_tagesschau-4701	0	3	7
12September_2009_Saturday_tagesschau-6591	0	0	10
04October_2010_Monday_heute-5954	0	3	7
06October_2010_Wednesday_tagesschau-640	0	3	7
08December_2009_Tuesday_heute-4213	2	4	4
16April_2010_Friday_tagesschau-6577	0	0	10
24November_2011_Thursday_heute-223	0	2	8
25August_2010_Wednesday_heute-6009	1	1	8
30August_2011_Tuesday_heute-783	0	1	9
10December_2011_Saturday_tagesschau-5420	0	2	8
03October_2012_Wednesday_tagesschau-7655	0	3	7
25November_2010_Thursday_tagesschau-2536	0	1	9
24January_2011_Monday_tagesschau-3712	0	6	4
02November_2010_Tuesday_heute-2754	1	0	9
27August_2009_Thursday_tagesschau-3285	0	1	9
10August_2010_Tuesday_tagesschau-3865	0	1	9
23April_2010_Friday_tagesschau-3667	0	3	7
03September_2010_Friday_tagesschau-8432	0	1	9
10November_2009_Tuesday_tagesschau-7250	0	1	9
24September_2009_Thursday_heute-6077	0	2	8
16September_2010_Thursday_tagesschau-6271	0	0	10
09January_2010_Saturday-1919	0	0	10
Total	4	44	202
Proportion	1.6 %	17.6 %	80.8 %

servations, we determine $n = 3, \gamma = 0.00001$ as the default configuration.

4. Human Evaluations

Given that existing metrics rely heavily on back-translation models, we conducted a human evaluation study to more comprehensively assess the generation quality. Since it is difficult to recruit sign language experts for evaluation, we adopt a more affordable and feasible crowdsourced approach: each volunteer is shown the ground-truth pose video and asked to select the most similar video among anonymized results generated by different models. We then count the number of votes received by each method. Specifically, we randomly sample 25 sentences (from the dev and test sets of PHOENIX14T) and generate the corresponding pose videos using different models, followed by anonymization of all videos. We recruit 10 participants, and each participant receives 25 sets of samples along with the following instruction: *"Please choose the video where the signer's actions appear most similar to those in the ground-truth video."*

As shown in Fig. 20, we report the detailed vote distribution of each model for every sample, as well as the final vote counts. From the per-sample voting results, FGDM exhibits a clear advantage on nearly all samples, except for 08December_2009_Tuesday_heute-4213 and 24January_2011_Monday_tagesschau-3712, where the vote difference between Sign-IDD and FGDM is relatively small. In terms of the overall vote count, FGDM receives 202 votes (80.8%), substantially higher than both PT (4 votes) and Sign-IDD (44 votes), demonstrating that FGDM delivers superior perceptual quality in human evaluation. Overall, the human evaluation results further validate the superior generation quality of FGDM.

5. Preliminary: Diffusion Model

Following [2, 6, 11, 13], we briefly review diffusion models, which consist of two main processes: diffusion and denoising.

Diffusion process. The diffusion process q produces a sequence of noisy samples $X_1, X_2, \ldots, X_T$ by progressively adding Gaussian noise to the original signal X_0 at each timestep $t \in [0, T]$, formulated as:

$$\begin{aligned} q(X_{1:T}|X_0) &:= \prod_{t=1}^{T} q(X_t|X_{t-1}), \\ q(X_t|X_{t-1}) &:= \mathcal{N}(X_t; \sqrt{1-\beta_t}X_{t-1}, \beta_t \boldsymbol{I}), \end{aligned} \tag{20}$$

where β_t denotes the noise variance schedule [11], and $\boldsymbol{I}$ is the identity matrix. Following Ho *et al.* [6], Eq. 20 can be reformulated to directly obtain X_t from X_0 without iteration:

$$\begin{aligned} q(X_t|X_0) &:= \mathcal{N}(X_t; \sqrt{\bar{a}_t}X_0, (1-\bar{a}_t)\boldsymbol{I}) \\ &:= \sqrt{\bar{a}_t}X_0 + \epsilon\sqrt{1-\bar{a}_t}, \epsilon \sim \mathcal{N}(0, \boldsymbol{I}), \end{aligned} \tag{21}$$

where $\bar{a}_t := \prod_{s=0}^{t} a_s$ and $a_s := 1 - \beta_s$. ϵ denotes Gaussian noise. With a sufficiently large T and a proper noise schedule β_t, $q(X_T)$ approximates an isotropic Gaussian distribution $\mathcal{N}(0, \boldsymbol{I})$.

Denoising process. The reverse process p, parameterized by θ, can be implemented in two ways: iteratively sampling $p_\theta(X_{t-1} \mid X_t)$ until X_0, or directly predicting X_0 via $p_\theta(X_0 \mid X_t)$.

In the first case, the posterior $q(X_{t-1} \mid X_t, X_0)$ can also be expressed as a Gaussian distribution derived from Bayes' theorem:

$$q(X_{t-1}|X_t, X_0) = \mathcal{N}(X_{t-1}; \tilde{\mu}(X_t, X_0), \tilde{\beta}_t \boldsymbol{I}), \tag{22}$$

where

$$\begin{aligned} \tilde{\beta}_t &:= \frac{1-\bar{a}_{t-1}}{1-\bar{a}_t}\beta_t, \\ \tilde{\mu}_t(X_t, X_0) &:= \frac{\sqrt{\bar{a}_{t-1}}\beta_t}{1-\bar{a}_t}X_0 + \frac{\sqrt{a_t}(1-\bar{a}_{t-1})}{1-\bar{a}_t}X_t \end{aligned} \tag{23}$$

are the variance and mean of the Gaussian distribution, respectively. However, Eq. 22 requires the unknown X_0 to compute the posterior. Thus, it can approximate $q(X_{t-1} \mid X_t, X_0)$ using

$$p_\theta(X_{t-1}|X_t) := \mathcal{N}(X_{t-1}; \mu_\theta(X_t, t), \Sigma_\theta(X_t, t)), \tag{24}$$

where

$$\Sigma_\theta(X_t,t)=\tilde{\beta}_t\boldsymbol{I},$$
$$\mu_\theta(X_t,t)=\frac{1}{\sqrt{a_t}}\left(X_t-\frac{\beta_t}{\sqrt{1-\bar{a}_t}}\epsilon_\theta(X_t,t)\right), \tag{25}$$

$\epsilon_\theta(X_t,t)$ denotes the noise predicted by the neural network, trained under supervision from $\epsilon\sim\mathcal{N}(0,\boldsymbol{I})$. In contrast, the second formulation directly predicts the clean data using a learned network $f_\theta(X_t,t)$, which is supervised by the original signal X_0.

The main difference between the two cases lies in the prediction target of the network: the first predicts the noise ϵ at each timestep, while the second predicts the original signal X_0. Since the latter offers better interpretability and is more suitable for computing regression losses, we adopt this formulation.

6. Preliminary: Graph Convolution Networks

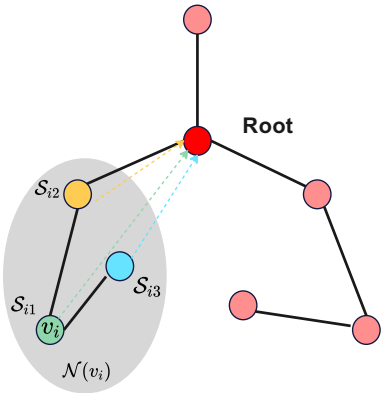


Figure 6. Illustration of the mapping strategy (Spatial Separation) in [20]. Different colors denote different subsets.

Since the proposed ASGCN is mainly inspired by ST-GCN [20] and AGCN [14], we briefly review these two representative works to better highlight our innovations.

Definition of ST-GCN. A human skeleton can be naturally modeled as a graph, where each joint corresponds to a vertex and each bone defines an edge connecting two adjacent joints. For the i-th vertex v_i, the graph convolution operation is formulated as: [20]:

$$X_{out}(v_i)=\sum_{v_j\in\mathcal{N}(v_i)}\frac{1}{Z_i(v_j)}X_{in}(v_j)\cdot w(l_i(v_j)), \tag{26}$$

where X_{in} denotes the feature map. $\mathcal{N}(v_i)$ denotes all 1-distance neighboring vertices (v_j) of the target vertex (v_i), which are involved in the convolution computation. w is a weighting function analogous to that in standard convolution, which generates a weight vector for each input. Note that, unlike conventional convolution with a fixed number of weight vectors, the number of vertices in $\mathcal{N}(v_i)$ varies dynamically.

To map each vertex with a unique weight vector, a mapping function l_i is designed specially in [20]. Specifically, Fig. 6 illustrates this strategy, where the green joint represents v_i and the gray region indicates its neighborhood $\mathcal{N}(v_i)$. The strategy empirically sets the kernel size as 3 and naturally divides $\mathcal{B}_i$ into 3 subsets: $\mathcal{S}_{i1}$ denotes the vertex itself (the green joint in Fig. 6); $\mathcal{S}_{i2}$ is the centripetal subset, consisting of neighboring joints closer to the gravity center (yellow joint); and $\mathcal{S}_{i3}$ is the centrifugal subset, containing neighbors farther from the gravity center (blue joint). $Z_i(v_j)$ represents the cardinality of the subset $\mathcal{S}_{ik}$ that includes v_j, serving to balance the contributions among different subsets.

Implementation of ST-GCN [20]. Typically, the input feature of a pose sequence, X_{in}, is represented as a tensor of shape $C\times S\times J$, where J denotes the number of joints (or vertices), S is the temporal length, and C is the number of channels. The spatial graph convolution in ST-GCN is applied with shared parameters across frames, and Eq. 26 can thus be reformulated as:

$$\mathbf{X}_{out}=\sum_{k}^{K_v}\mathbf{W}_k\cdot\mathbf{X}_{in}(\mathbf{A}_k\odot\mathbf{M}_k), \tag{27}$$

where K_v denotes the kernel size in the spatial domain, which is set to 3 based on the partition strategy above. The adjacency matrix for the k-th subset is defined as:

$$\mathbf{A}_k=\mathbf{\Lambda}_k^{-\frac{1}{2}}\bar{\mathbf{A}}_k\mathbf{\Lambda}_k^{-\frac{1}{2}}, \tag{28}$$

where $\bar{\mathbf{A}}_k\in\mathbb{R}^{J\times J}$ is the subset-specific adjacency matrix, and each element $\bar{\mathbf{A}}_k^{ij}$ indicates whether vertex v_j belongs to the subset $\mathcal{S}_{ik}$ of vertex v_i. The normalization matrix is given by

$$\mathbf{\Lambda}_k^{ii}=\sum_j\bar{\mathbf{A}}_k^{ij}+\alpha, \tag{29}$$

where $\alpha=0.001$ avoids empty rows. $\mathbf{W}_k\in\mathbb{R}^{C_{out}\times C_{in}\times 1\times 1}$ represents the learnable weights of the 1×1 convolution, corresponding to the weighting function w in Eq. 26. $\mathbf{M}_k\in\mathbb{R}^{J\times J}$ denotes the attention mask that measures the importance of each vertex, and $\odot$ indicates the dot product.

Implementation of AGCN [14]. The key improvement of AGCN lies in decomposing the adjacency matrix into three parts: $\mathbf{A}_k$, $\mathbf{B}_k$, and $\mathbf{C}_k$. Specifically, $\mathbf{A}_k$ corresponds to the fixed topology used in ST-GCN, $\mathbf{B}_k$ is a learnable matrix of size $J\times J$, and $\mathbf{C}_k$ models global correlations and is formulated as:

$$\mathbf{C}_k=softmax(\mathbf{X}_{in}^\top\mathbf{W}_{\theta k}^\top\mathbf{W}_{\phi k}\mathbf{X}_{in}), \tag{30}$$

where $\mathbf{W}_{\theta k}$ and $\mathbf{W}_{\phi k}$ are the learnable parameters of the embedding functions θ and ϕ, respectively. Compared with

ST-GCN, AGCN can learn the latent relationships between physically non-adjacent joints through the learnable matrix $\mathbf{B}_k$, while $\mathbf{C}_k$ dynamically adapts to the input features, enabling more flexible and data-driven graph connectivity.

Our improvement. Conventional graph convolution methods typically share convolutions across frames, which is reasonable for tasks such as action recognition, where the goal is to model overall motion categories and maintain temporal consistency. However, in sign language sequences, the semantic role of each joint may vary over time, and local spatial dependencies can dynamically change across frames. To better adapt to the complex and dynamic nature of sign language, we refine the graph convolution as follows:

$$X^i_{out} = \mathbf{W}_\theta \cdot \sum_{k=1}^{K_v} \mathbf{W}_{\delta k} \odot \left(X^i_{in} \cdot \left((\mathbf{A}^i_{ak} + \mathbf{A}^i_{bk}) \odot \mathcal{M}^i_k \right) \right). \tag{31}$$

Here, we decompose $\mathbf{W}$ in Eq. 27 into $\mathbf{W}_\theta$ and $\mathbf{W}_\delta$, effectively reducing the number of parameters from $C_{in}(K \cdot C_{out} - K - C_{out})$. In addition, our graph convolutions are designed to be frame-specific rather than shared across frames. This design enables $\mathbf{A}^i_a$ to focus on aggregating correlations among local neighbors, thereby avoiding interference from distant or semantically irrelevant joints. The topology-based matrix $\mathbf{A}^i_b$, shared across frames, preserves consistent skeletal topology and provides a stable structural prior. Meanwhile, we replace $\mathbf{M}_k$ in Eq. 27 with the semantic mask $\mathcal{M}^i_k$, which adaptively incorporates linguistic conditions, allowing each frame's pose representation to better align with the semantic stage of its corresponding gloss. Finally, our ASGCN significantly outperforms previous GCNs on the SLP task, as shown in Tab. 6.

7. Preliminary: Connectionist Temporal Classification

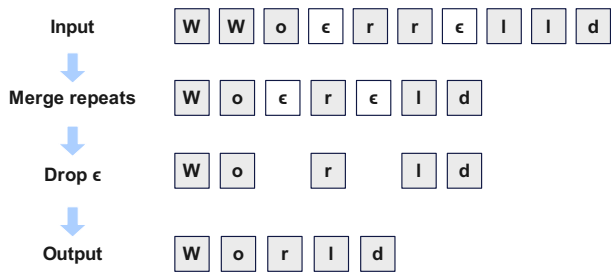


Figure 7. Illustration of the many-to-one mapping function $\mathcal{B}$ in CTC, where ε denotes the $blank$ label.

CTC [3] is an algorithm designed for sequence-to-sequence learning, particularly suitable for tasks where the input and output lengths differ and no frame-level alignment annotations are available. Since our proposed SCG loss is built upon CTC, we briefly review its core concept here.

Core idea of CTC. CTC introduces an additional $blank$ label to represent unlabeled segments (such as movement epenthesis or non-gesture intervals in sign language). Accordingly, the original vocabulary set $\mathbb{G}$ is extended to $\mathbb{G}' = \mathbb{G} \cup \{blank\}$. CTC then defines a many-to-one mapping function $\mathcal{B} : \mathbb{G}'^T \to \mathbb{G}^{\leq T}$ to align the path sequence $\pi \in \mathbb{G}'^T$ with the target labeling $\boldsymbol{l} \in \mathbb{G}^{\leq T}$ by sequentially removing repeated labels and blank symbols. As shown in Fig. 7, we take the output "World" as an example to illustrate this collapsing process.

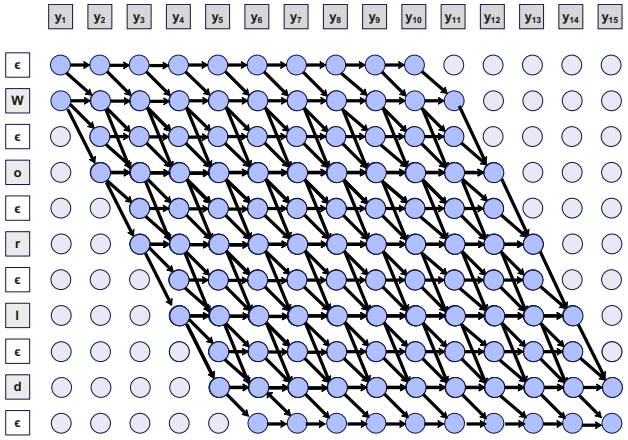


Figure 8. Visualization of all feasible alignment paths, i.e., $\mathcal{B}^{-1}(\boldsymbol{l})$, that can be collapsed into the target label sequence.

With the help of this function, CTC provides supervision by summing the probabilities of all feasible alignment paths:

$$\mathcal{L}_{\text{CTC}} = -\log \sum_{\pi \in \mathcal{B}^{-1}(\boldsymbol{l})} p(\pi \mid y^o), \tag{32}$$

where $\mathcal{B}^{-1}(\boldsymbol{l})$ denotes the set of all alignments π of length S' that can be collapsed into $\boldsymbol{l}$. These feasible paths can be efficiently computed using a dynamic programming algorithm:

$$\alpha_{(t,s)} = \begin{cases} \left[\alpha_{(t-1,s)} + \alpha_{(t-1,s-1)}\right] p(y_t = l_s), \\ \text{if } l_s = \textit{blank or } l_s = l_{s-2}, \\ \left[\alpha_{(t-1,s)} + \alpha_{(t-1,s-1)} + \alpha_{(t-1,s-2)}\right] p(y_t = l_s), \\ \text{otherwise.} \end{cases} \tag{33}$$

Here, $\alpha_{(t,s)}$ represents the accumulated probability of aligning to the s-th label at time step t. As illustrated in Fig. 8, all feasible alignment paths corresponding to the label sequence "World" can be obtained through this dynamic programming procedure. The probability of each alignment path π is computed under the conditional independence assumption across time steps:

$$p(\pi \mid y^o) = \prod_{s=1}^{S'} p(\pi_s \mid y^o), \tag{34}$$

Algorithm 1 FGDM Training

```
def training_loss(gls_seq, trg_seq, T):
  # gls_seq: [B, L], trg_seq: [B, S, J, 3]
  # T: maximum number of timesteps
  # B: batch size, L: gloss count
  # S: frame count, J: joint count


  # Corrupt trg_seq
  t = randint(0, T)               # timestep
  noise = normal(mean=0, std=1) # noise: [B, S, J, 3]
  alpha_cp = alpha_cumprod(t)
  trg_crpt = sqrt(    alpha_cp) * trg_seq +
             sqrt(1 - alpha_cp) * noise

  # trg_pred: denoised prediction
  # trg_pred_mid: intermediate feature
  trg_pred, trg_pred_mid = FGDM_denoiser(trg_crpt,
      gls_seq, t)

  # Set regression loss
  loss_reg = MAE(trg_pred, trg_seq) +
  lamda * MSE(bone(trg_pred), bone(trg_seq))

  # Set SCG loss
  loss_scg = SCG(trg_pred_mid, gls_seq, t)

  # Set total loss
  loss_total = loss_reg + gamma * loss_scg

  return loss_total
```

Algorithm 2 FGDM Inference

```
def inference(gls_seq, T, I):
  # gls_seq: [B, L], T: maximum number of timesteps
  # I: number of iterations

  # Initialize noisy target poses: [B, S, J, 3]
  trg_t = normal(mean=0, std=1)

  # Sample timesteps uniformly
  times = reversed(linespace(0, T, I + 1))

  # [(T*(1-i/I), T*(1-(i+1)/I))], i = 0,...,I-1
  time_pairs = list(zip(times[:-1], times[1:])

  for t_now, t_next in zip(time_pairs):

    # Predict trg_0 from trg_t
    trg_0 = FGDM_denoiser(trg_t, gls_seq, t_now)

    if t_next!=0:

       # Estimate trg_t at t_next
       trg_t = ddim_step(trg_t, trg_0, t_now, t_next)

  return trg_0
```

where the per-frame probability $p(\pi_s \mid y^o)$ is obtained via a softmax operation over the output logits y^o.

Our improvement. The SCG loss proposed in this work is an improved variant of Eq. 32, where a time-aware term $\frac{1}{e^{\alpha \frac{t}{T}}}$ is introduced to better adapt the training process of the diffusion model and facilitate stable convergence of the Semantic Decoder. As shown in Tab. 7, the model trained with the proposed SCG loss achieves significantly better performance than that using the original CTC loss.

8. Clarification of Potential Issues

Why not include non-manual (e.g., facial) features? Our method can also be extended to generate non-manual features by tuning a few hyperparameters and training on datasets that include them. However, most existing SLP works [12, 16–19] still focus on manual features, since non-manual ones usually involve many keypoints but carry limited semantic information (e.g., emotion). Although incorporating non-manual features may slightly improve back-translation metrics (e.g., WER, BLEU), it would make comparisons with manual-only approaches less fair. Therefore, we focus on advancing the methodology itself rather than broadening the feature scope.

9. Details of Architectures

As shown in Tab. 21 and Tab. 22, we present the detailed architectures of FGDM and SCG, including the name of each component, its output shape, and the configuration of the stacked modules with corresponding parameters.

10. Algorithm

In this section, we present the training and inference processes of FGDM in algorithmic form for clearer understanding.

FGDM Training. Alg. 1 presents the pseudocode of FGDM training. Specifically, a timestep t is randomly sampled, and the target sequence trg_seq is noised t times (as in Eq. 21) to obtain the corrupted sequence trg_crpt. The FGDM denoiser then predicts the target sequence trg_pred conditioned on the gloss sequence gls_seq and t. The regression loss is computed between trg_pred and trg_seq, while the intermediate output trg_pred_mid is used with gls_seq to calculate the SCG loss. The final training objective is the weighted sum of the regression and SCG loss.

FGDM Inference. Alg. 2 presents the pseudocode of FGDM inference. First, Gaussian noise trg_t is randomly sampled as the initial noisy sequence. The total number of iterations I determines the timesteps used for each iteration. At each iteration, FGDM predicts trg_0 based on the noisy input trg_t, the gloss condition gls_seq, and the current timestep t_now. The DDIM scheduler then uses trg_0 to compute the input noise for the next iteration. After all iterations, trg_0 is taken as the final output of the model.

11. Details of Evaluation Metrics

This section will provide a detailed description of the evaluation metrics, including WER, BLEU, ROUGE, and FID.

WER (Word Error Rate). During evaluation, the generated sign poses are recognized into gloss sequences, and WER is used to measure the difference between the predicted and ground-truth gloss sequences. Specifically, WER quantifies the minimum number of substitutions (#sub), insertions (#ins), and deletions (#del) required to align the

name	output size	structure
Embedding	(16,S,J)	Conv (in_channels=16, out_channels=16,kernel_size=1×1)
Gloss Encoder	(L,512)	[Self-MHA (heads=4, hidden_size=512) Self-MHA (heads=4, hidden_size=512) FFN (hidden_size=2048)] ×2
Focal stage	(16,S,J)	[ASGCN (in_channels=16, out_channels=16) TCN (in_channels=16, out_channels=16, kernel_size=9)] ×3
F2G_transform	(S,512)	Linear (input_dims=16*J,output_dims=512)
General stage	(S,512)	[Self-MHA (heads=4, hidden_size=512) Cross-MHA (heads=4, hidden_size=512) FFN (hidden_size=2048)] ×2
Classifier	(S, J*3)	[LayerNorm (input_dims=512) Linear (input_dims=512, output_dims=J*3)]

Table 21. Main architectures for Focal General Diffusion Network (FGDM).

name	output size	structure
V2S_Adapter	(S,512)	[Linear (input_dims=512,output_dims=128) ReLU Linear (input_dims=128,output_dims=512)] ×2
LSD	(S',1024)	[TCN (in_channels=512, out_channels=1024, kernel_size=5) Maxpool (kenel_size=2, strides=2) TCN (in_channels=16, out_channels=16, kernel_size=5)] ×3
GSD	(S',1024)	BiLSTM (input_size=1024, hidden_size=512, output_size=1024)
Gloss_Classifier	(S', num_glosses+1)	Linear (input_dims=1024,output_dims=num_glosses+1)

Table 22. Main architectures for Semantic Consistent Guidance (SCG).

predicted sequence with the reference (#ref), and is computed as

$$\mathrm{WER} = \frac{\#\mathrm{sub} + \#\mathrm{ins} + \#\mathrm{del}}{\#\mathrm{ref}}. \tag{35}$$

A **lower** WER indicates **better** recognition accuracy.

BLEU (BiLingual Evaluation Understudy). During evaluation, the generated poses are translated into spoken language, and BLEU is used to assess translation accuracy. BLEU measures the n-gram overlap between the predicted and reference sentences, reflecting their lexical and sequential consistency. BLEU-N is defined as:

$$\text{BLEU-N} = \mathrm{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right), \tag{36}$$

where w_n denotes the weight assigned to each n-gram, and BP is the brevity penalty used to penalize overly short predictions, defined as:

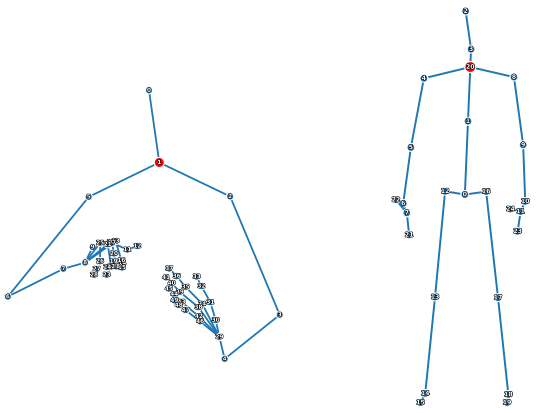
$$\mathrm{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \le r \end{cases}. \tag{37}$$

where c and r represent the lengths of the candidate and reference translations, respectively. The n-gram precision p_n is computed as:

$$p_n = \frac{\sum_{C \in \text{Candidates}} \sum_{\text{n-gram} \in C} \mathrm{Count}_{clip}(\text{n-gram})}{\sum_{C' \in \text{Candidates}} \sum_{\text{n-gram'} \in C'} \mathrm{Count}(\text{n-gram'})}, \tag{38}$$

$\mathrm{Count}_{clip}(\text{n-gram})$ denotes the number of occurrences of an n-gram in the reference sequence, while $\mathrm{Count}(\text{n-gram'})$ represents the count of n-gram' in the candidate sequence. The numerator thus measures how many n-grams from the candidate appear in the reference, and the denominator represents the total number of n-grams in all candidate sequences. We evaluate using BLEU-1, BLEU-2, BLEU-3, and BLEU-4, where higher scores indicate more accurate generation.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation). We use ROUGE-L [9], a metric from the ROUGE family, to further evaluate the quality of the back-translated sentences. ROUGE-L measures the similarity between a candidate sentence and its reference by comput-



(a) PHOENIX14T (b) USTC-CSL

Figure 9. Skeleton topology details of the PHOENIX14T (a) and USTC-CSL (b), where the red joint denotes the root joint.

ing the length of their longest common subsequence (LCS). Given a reference sentence $R = (r_1, r_2, \ldots, r_m)$ and a candidate sentence $C = (c_1, c_2, \ldots, c_n)$, the length of their longest common subsequence is denoted as $LCS(R, C)$. Based on this, the recall, precision, and F-measure are defined as:

$$R_{LCS} = \frac{LCS(R,C)}{m}, \quad P_{LCS} = \frac{LCS(R,C)}{n}, \tag{39}$$

$$F_{LCS} = \frac{(1+\beta^2) \cdot R_{LCS} \cdot P_{LCS}}{R_{LCS} + \beta^2 \cdot P_{LCS}}, \tag{40}$$

where m and n are the lengths of the reference and candidate sentences, respectively, and β is used to balance recall and precision (we set to 1.2). A higher ROUGE-L score indicates better generation quality.

FID (Fréchet Inception Distance). The Fréchet Inception Distance (FID) [5] is widely used to evaluate the quality of generated images or sequences by comparing the distributions of features extracted from real and generated data. Let $\mathcal{N}(\mu_r, \Sigma_r)$ and $\mathcal{N}(\mu_g, \Sigma_g)$ denote the multivariate Gaussian distributions fitted to the feature representations of real and generated samples, respectively. FID is defined as:

$$\text{FID} = |\mu_r - \mu_g|_2^2 + \text{Tr}\Big(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}\Big), \tag{41}$$

where μ_r, Σ_r are the mean and covariance of features from real data, and μ_g, Σ_g are the corresponding statistics for generated data. A lower FID score indicates that the generated distribution is closer to the real one, reflecting higher-quality and more realistic samples.

12. Additional Data Processing Details

This section mainly provides additional details on data processing for better reproducibility, and the corresponding data processing code will be released as open source.

PHOENIX14T. Since PHOENIX14T does not provide any keypoints, we extract them using OpenPose [1]. For each frame, we obtain 50 keypoints, including the first 8 from BODY_25 and 21 keypoints for each hand. The keypoint indices and topology are shown in Fig. 9a. Multithreading is used to accelerate extraction. Subsequently, a skeletal correction model [21] is applied to refine the keypoints and convert 2D coordinates into 3D coordinates. All coordinates are divided by 3 for scaling and rounded to five decimal places. Finally, each frame contains 150 (50×3) joint values followed by a counter value, all separated by spaces. Each sequence is separated by a new line.

USTC-CSL. The USTC-CSL dataset [4] provides 3D keypoint coordinates captured using Kinect 2.0, where each frame contains 25 joints. The joint topology and encoding format are illustrated in Fig. 9b. We apply Z-score normalization to the raw data and fix the root joint coordinates. All coordinates are further divided by 3 as a scaling factor, retaining five decimal places. Each frame contains 75 (25×3) joint values followed by a counter value, all separated by spaces, and each sequence is separated by a newline. Since each signer repeats each sentence five times, we randomly select one instance per sentence when splitting the dataset.

13. Visualizations

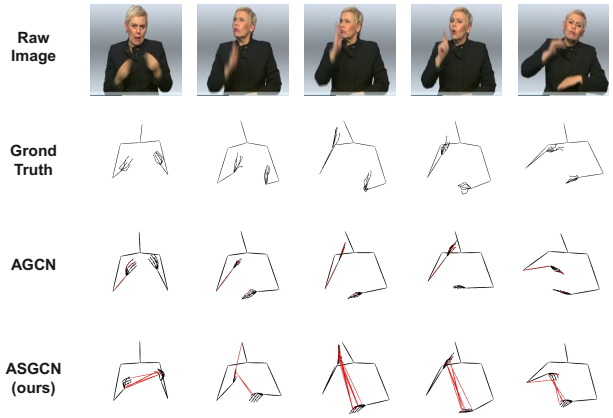


Figure 10. Comparison of the most attended joint connections across different frames between AGCN [14] and ASGCN. Red lines indicate the connections dynamically emphasized by the model.

Comparison of the most attended joint connections. As shown in Fig. 10, we compare the top 10 most attended joint connections of ASGCN and AGCN [14] across different frames. Both models primarily focus on hand-related connections while paying less attention to the torso. However, AGCN places more attention on the right hand and largely ignores the left hand and inter-hand relationships. In contrast, ASGCN distributes attention more evenly between both hands, effectively capturing their interactions.

Moreover, ASGCN adjusts its focus dynamically across frames—for example, in the second and last columns, the model shifts attention in response to head and arm movements—demonstrating strong frame-wise adaptability. Although AGCN removes the constraint of a fixed topology, it fails to adapt its focus across frames and cannot leverage semantic conditions or neighborhood correlations, which greatly limits its flexibility. Overall, this visualization highlights the strong frame-wise adaptability of ASGCN.

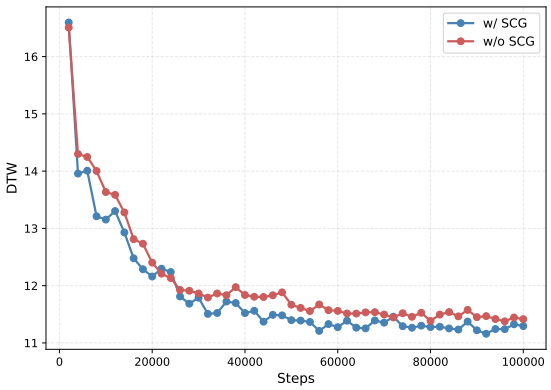


Figure 11. Comparison of DTW variation with and without SCG (w/ SCG vs. w/o SCG) over training steps.

Comparison of DTW variation with and without SCG. Dynamic Time Warping (DTW) measures the temporal consistency between the generated pose sequence and the corresponding ground-truth, where a lower value indicates higher consistency. As shown in Fig. 11, we compare the DTW variations during training with and without SCG. When SCG is applied, the DTW values remain consistently lower, indicating that SCG enhances the temporal alignment between the generated and ground-truth sequences (i.e., more natural motion rhythm and timing). This improvement arises because SCG aligns the generated feature and gloss sequences in the semantic space, which accelerates convergence and promotes the generation of temporally coherent poses.

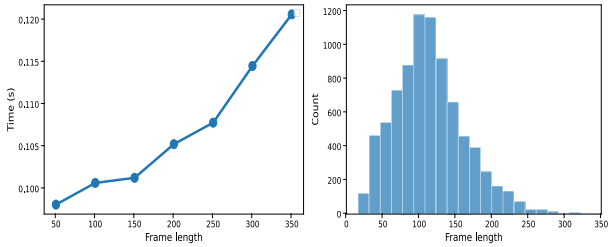


Figure 12. Left: Inference time vs. sequence length. Right: Distribution of sequence lengths in the PHOENIX14T dataset.

Investigation of real-time inference. As shown in the left panel of Fig. 12, we analyze the effect of the generated pose sequence length on the inference time of FGDM. As the sequence length increases from 50 to 350 frames, the inference time rises almost linearly from 0.100 s to 0.120 s. Although the time increases with sequence length, FGDM maintains real-time performance within this range. The right panel illustrates the frame-length distribution in the PHOENIX14T dataset, where most samples fall within 50–200 frames, and only a few exceed 300 frames but none go beyond 350. This demonstrates that FGDM can complete generation within 0.120 s for all typical sequence lengths, ensuring real-time interaction capability.

Qualitative comparison with previous methods on PHOENIX14T and USTC-CSL. As shown in Fig. 13, we compare the proposed FGDM with previous methods (PT [12] and Sign-IDD [17]). The upper examples show results on the PHOENIX14T dataset, while the lower ones are from USTC-CSL. PHOENIX14T contains a large number of hand keypoints with rich fine-grained details. Compared to PT and Sign-IDD, FGDM reconstructs these details more accurately, exhibiting fewer hand deformations and more natural poses. Moreover, FGDM can still reconstruct poses reasonably well even when the ground truth is corrupted by motion blur. In contrast, USTC-CSL includes fewer joints and limited hand-related information, where the focus lies on the accuracy of overall body poses. Although its simpler skeletal connectivity may weaken the effect of the Focal stage, FGDM still generates poses that are closer to the ground truth, particularly in the natural bending and interaction between the two hands. The qualitative comparisons on both datasets demonstrate that FGDM effectively captures fine-grained spatial dependencies and maintains robust generation quality under varying skeletal complexities.

14. Limitations and Future Works

While FGDM achieves strong performance in both quantitative and qualitative evaluations, several limitations remain. 1) Computational overhead. The two-stage architecture (Focal and General) improves generation quality but introduces slight computational overhead compared to single-stage models. Although inference remains real-time (0.12 s for sequences up to 350 frames), further optimization could enhance scalability for longer or more complex sign sequences. 2) Motion artifacts. The generated pose sequences occasionally exhibit hand size variations, jitter, or brief pauses between frames. These artifacts are insufficiently constrained during training, and no established metrics currently exist to evaluate such temporal or spatial inconsistencies. Future work could incorporate motion refinement modules, temporal consistency losses, and perceptual metrics for assessing motion realism in sign language generation. 3) Dataset limitations. Although experiments were conducted on two standard benchmarks, their limited vocabulary and scale do not fully capture real-world sign language diversity. In the future, we plan to expand the dataset

through large-scale data collection and move toward developing production-level SLP systems with stronger generalization and robustness. 4) Evaluation limitations. Current SLP evaluation relies heavily on back-translation. In this work, we observe for the first time that predicted poses can even outperform the ground truth under this metric, indicating that back-translation may become unreliable as models continue to advance. Although we additionally conduct human evaluation to validate the results, this process is time-consuming and labor-intensive, especially when a large number of samples must be assessed. Addressing these limitations will require the development of more robust back-translation models and the design of evaluation protocols that more accurately reflect human-perceived quality.

References

[1] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299, 2017. 10

[2] Shoufa Chen, Peize Sun, Yibing Song, and Ping Luo. Diffusiondet: Diffusion model for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 19830–19843, 2023. 5

[3] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006. 3, 7

[4] Dan Guo, Wengang Zhou, Houqiang Li, and Meng Wang. Hierarchical lstm for sign language translation. In *Proceedings of the AAAI conference on artificial intelligence*, 2018. 10

[5] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 10

[6] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 5

[7] Wencan Huang, Wenwen Pan, Zhou Zhao, and Qi Tian. Towards fast and high-quality sign language production. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 3172–3181, 2021. 2

[8] Kazuya Kawakami. *Supervised sequence labelling with recurrent neural networks*. PhD thesis, Ph. D. thesis, 2008. 3

[9] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004. 9

[10] Yuecong Min, Aiming Hao, Xiujuan Chai, and Xilin Chen. Visual alignment constraint for continuous sign language recognition. In *proceedings of the IEEE/CVF international conference on computer vision*, pages 11542–11551, 2021. 3

[11] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021. 5

[12] Ben Saunders, Necati Cihan Camgoz, and Richard Bowden. Progressive transformers for end-to-end sign language production. In *European Conference on Computer Vision*, pages 687–705. Springer, 2020. 2, 5, 8, 11

[13] Wenkang Shan, Zhenhua Liu, Xinfeng Zhang, Zhao Wang, Kai Han, Shanshe Wang, Siwei Ma, and Wen Gao. Diffusion-based 3d human pose estimation with multi-hypothesis aggregation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14761–14771, 2023. 5

[14] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12026–12035, 2019. 1, 2, 6, 10

[15] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with multi-stream adaptive graph convolutional networks. *IEEE Transactions on Image Processing*, 29:9532–9545, 2020. 1, 2

[16] Shengeng Tang, Richang Hong, Dan Guo, and Meng Wang. Gloss semantic-enhanced network with online back-translation for sign language production. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 5630–5638, 2022. 2, 8

[17] Shengeng Tang, Jiayi He, Dan Guo, Yanyan Wei, Feng Li, and Richang Hong. Sign-idd: Iconicity disentangled diffusion for sign language production. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7266–7274, 2025. 2, 5, 11

[18] Shengeng Tang, Feng Xue, Jingjing Wu, Shuo Wang, and Richang Hong. Gloss-driven conditional diffusion models for sign language production. *ACM Transactions on Multimedia Computing, Communications and Applications*, 21 (4):1–17, 2025.

[19] Pan Xie, Qipeng Zhang, Peng Taiying, Hao Tang, Yao Du, and Zexian Li. G2p-ddm: Generating sign pose sequence from gloss sequence with discrete diffusion model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6234–6242, 2024. 8

[20] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI conference on artificial intelligence*, 2018. 1, 2, 6

[21] Jan Zelinka and Jakub Kanis. Neural sign language synthesis: Words are our glosses. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3395–3403, 2020. 10

[22] Albert Zeyer, Ralf Schlüter, and Hermann Ney. Why does ctc result in peaky behavior? *arXiv preprint arXiv:2105.14849*, 2021. 3

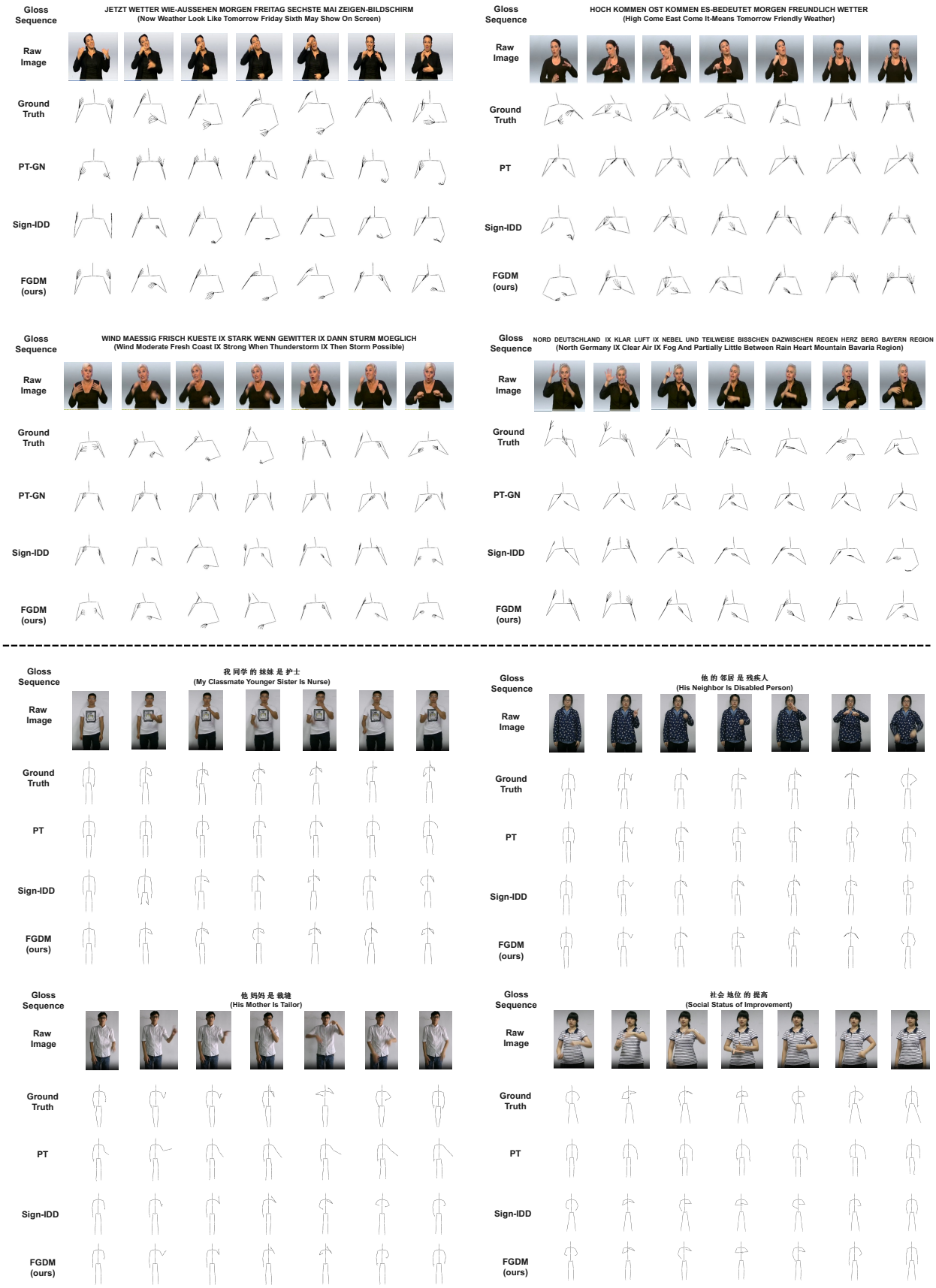


Figure 13. Qualitative comparison of generation quality against previous methods on PHOENIX14T (top) and USTC-CSL (bottom).