Traduzione automatica "neurale" per il ladino della Val Badia

Samuel Frontull, Georg Moser

1. Introduzione

La traduzione automatica "neurale" *(NMT: Neural Machine Translation)* è il metodo più avanzato di traduzione automatica. La capacità dei modelli neurali di individuare ed elaborare strutture linguistiche complesse e variazioni contestuali si è dimostrata estremamente efficace. Tuttavia, a causa dell'elevata richiesta di disponibilità di dati, l'approccio neurale è stato inizialmente inferiore ai metodi statistici nella traduzione di lingue a basse risorse. Motivati dalla necessità di superare barriere linguistiche e preservare il patrimonio culturale, negli ultimi anni sono stati sviluppati diversi metodi per applicare la NMT anche a lingue meno diffuse con risorse linguistiche limitate (cf. Ranathunga 2023, Haddow 2022, Shi 2022, Wang 2021).

In questa relazione presentiamo un approccio alla NMT per il ladino basato su metodi e su risultati riportati nella letteratura. Questo problema è particolarmente interessante perché, da un lato, è disponibile solo una quantità limitata di dati paralleli e, dall'altro, ci sono diverse varianti del ladino dolomitico (in Val Badia, Gardena, Fassa, Livinallongo e Ampezzo).

Ladinia XLVIII (2024), 119–144; DOI 10.54218/ladinia.48.119-144.
ISSN 1124–1004;

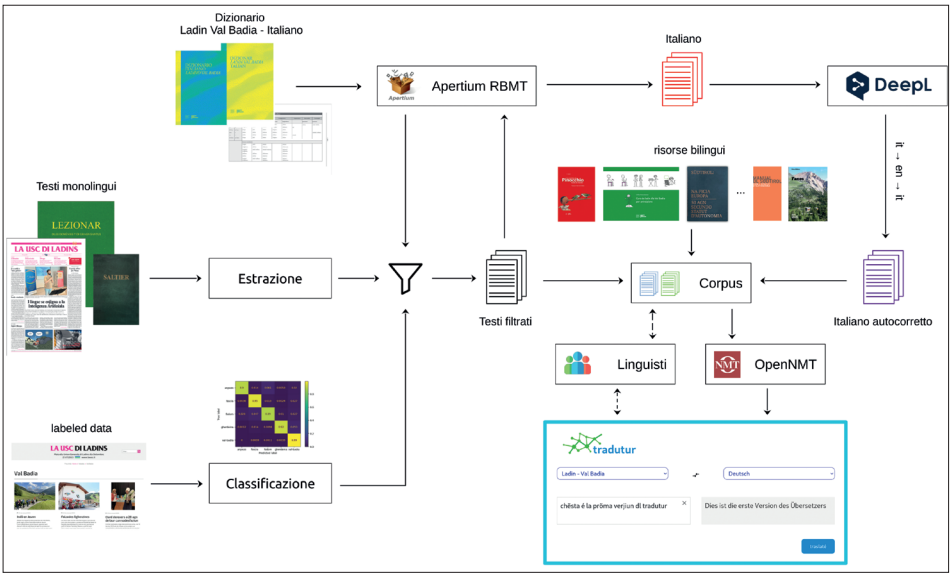

Fig. 1: Metodologia proposta.

Soprattutto per quanto riguarda la creazione di testi di traduzione ladini, è importante non confondere le rispettive varianti nei testi di addestramento. Ciò significa che i testi devono essere categorizzati e filtrati.[1] Inoltre, in una fase preliminare alla creazione del corpus, utilizziamo una traduzione basata su regole, che deve essere implementata separatamente per ogni variante.

Come fonte principale per i testi ladini ci basiamo principalmente sul giornale ladino "La Usc di Ladins".[2] Sfruttiamo l'efficienza della cosiddetta tecnica di *back-translation* (cf. Sennrich/Haddow/Birch 2016a) e addestriamo sui dati sintetizzati modelli *Transformer* da zero e, seguendo l'approccio del *transfer learning*, in un processo di *fine-tuning*. Poiché ciò richiede una catena di processo sofisticata, limitiamo i nostri esperimenti alla variante del ladino della Val Badia.

Per l'elaborazione dei testi ladini ci basiamo sul dizionario "Italiano–Ladin Val Badia" (Moling 2016). Questo dizionario è già stato digitalizzato e comprende non solo i singoli lemmi, ma anche le corrispondenti flessioni. Abbiamo importa-

[1] Per la variante della Val Badia la riforma ortografica del 2015 impone un'ulteriore elaborazione dei testi.

[2] <https://www.lausc.it>, [27/09/2023]; questa data di consultazione vale per tutti i siti indicati in questo contributo.

to tali dati in *Apertium* (cf. Forcada/Tyers 2016). Utilizziamo l'implementazione di *Apertium* per effettuare operazioni di filtraggio e pulizia dei dati, nonché per generare traduzioni preliminari. Successivamente, perfezioniamo ulteriormente le traduzioni generate utilizzando l'*API DeepL*, al fine di apportare un miglioramento soprattutto a livello grammaticale. Il corpus risultante costituisce la base per i vari esperimenti. La metodologia proposta è illustrata nella Figura 1.

Dimostriamo che questo approccio è un modo possibile per superare le varie sfide e che i modelli di traduzione producono traduzioni di qualità ragionevolmente elevata. Abbiamo creato un'applicazione web che offre accesso ai modelli di traduzione sviluppati: <https://tradutur-informatik.uibk.ac.at/>.

Per la revisione continua dei testi, abbiamo sviluppato una piattaforma che consente di apportare ulteriori correzioni e, di conseguenza, perfezionare i modelli.

La Sezione 2 illustra i lavori e approcci correlati. La Sezione 3 descrive il processo di acquisizione dei dati e la creazione del corpus. Nella Sezione 4 viene fornita un'analisi dei metodi utilizzati, degli esperimenti condotti e dei risultati ottenuti. Nella Sezione 5 presentiamo brevemente l'applicazione sviluppata per la revisione del corpus. Infine, nella Sezione 6 concludiamo con una sintesi e una discussione sul lavoro futuro.

2. Lavori correlati

Le possibilità di traduzione automatica per la lingua ladina sono già state studiate in Frontull/Hell 2022, ma esclusivamente valutando la traduzione automatica basata su regole e quella statistica. Il presente lavoro si fonda sulle basi stabilite nel lavoro citato, ma supera in modo significativo i risultati ottenuti in precedenza.

Nel nostro approccio sintetizziamo dati paralleli che poi raffiniamo ulteriormente. Un approccio analogo in due fasi è stato presentato in Pourdamghani 2019. La nostra metodologia si distingue per l'impiego di un sistema di traduzione basato su regole piuttosto che su una traduzione letterale. Inoltre, non utilizziamo un modello appositamente addestrato per migliorare le traduzioni, ma l'*API* di *DeepL*.

Il rischio della *back-translation* è quello di generare testi di scarsa qualità, con conseguente diminuzione della qualità della traduzione automatica. In un contesto a basse risorse, l'impatto è ancora più critico. È quindi importante non utilizzare

traduzioni sintetizzate in maniera ingenua. Utilizziamo il metodo presentato in Imankulova/Sato/Komachi 2017, che filtra i testi sintetizzati in base alla somiglianza (punteggio BLEU) delle loro traduzioni con l'originale. Tuttavia, nel processo di addestramento non distinguiamo tra traduzioni sintetizzate e traduzioni manuali, ad esempio mediante una marcatura (cf. Caswell/Chelba/Grangier 2019) o la ponderazione (cf. Dou/Anastasopoulos/Neubig 2020) dei testi. Queste tecniche devono ancora essere valutate.

In Fadaee/Bisazza/Monz 2017 è stato presentato un metodo alternativo per l'acquisizione di ulteriori dati d'addestramento. L'idea di base è quella di riutilizzare testi esistenti sostituendo le parole che vi ricorrono con parole che ricorrono di rado, migliorando così anche la stabilità dei modelli. I suggerimenti sono forniti da modelli linguistici appositamente addestrati per la lingua in questione. In questa relazione non trattiamo la tematica della stabilità. Tuttavia, l'approccio citato potrebbe essere approfondito in futuro se si intende lavorare in questa direzione, anche in combinazione con il dizionario su cui possiamo fare affidamento.

Nella NMT, la qualità finale della traduzione è influenzata da molti fattori. Un elemento importante è l'architettura del modello e il suo processo di addestramento in cui si possono impostare numerose configurazioni e parametri. Nei nostri esperimenti utilizziamo i *Transformer*. Consigli generici ed empirici per le loro configurazioni sono riportati in Popel/Bojar 2018. In contesti a risorse ridotte, le configurazioni adatte sono ancora più importanti, come dimostrato in Sennrich/Zhang 2019 e Araabi/Monz 2020. Nei nostri esperimenti seguiamo le raccomandazioni fornite in queste ricerche.

In questo lavoro, adattiamo modelli di traduzione liberamente disponibili tra lo spagnolo e l'italiano per la traduzione tra il ladino e l'italiano. Questo approccio ha stimolato ulteriori domande di ricerca, come ad esempio quali lingue siano più adatte per il processo del *fine tuning* (cf. Dabre/Fujita/Chu 2019) o quale sia il livello di adattamento che è necessario concedere (cf. Bapna/Firat 2019). L'architettura modulare *(encoder–decoder)* dei *Transformer* consentirebbe di riutilizzare l'*encoder* (codificatore) o il *decoder* (decodificatore) e di riaddestrare soltanto una componente. Si potrebbe decidere, ad esempio, di non modificare ulteriormente il decodificatore se la lingua di destinazione è già stata "allenata". In questo lavoro non abbiamo affrontato tali questioni e non fissiamo alcun parametro durante la fase del *fine tuning*, ma seguiamo un approccio semplice come quello descritto in Kocmi/Bojar 2018.

Grazie ai progressi nel settore della traduzione automatica delle lingue a basse risorse, sono stati sviluppati diversi sistemi di traduzione per lingue minoritarie basati sulla tecnologia neurale. *Meta AI* ha integrato con successo oltre 150 lingue a basse risorse nel suo progetto "No Language Left Behind" (cf. NLLB Team 2022). *Google* ha recentemente aggiunto più di 20 nuove lingue secondarie al suo traduttore (cf. Bapna 2022), e anche *Microsoft* sta portando avanti questa tematica da un po' di tempo (cf. Gu et al. 2018). Il progetto "GoURMET" (cf. van der Kreeft et al. 2022) ha avuto un grande successo nello sviluppo di un sistema di traduzione per 16 lingue a basse risorse. Sono disponibili nuovi strumenti di traduzione, ad esempio per il romancio[3], le lingue ugro-finniche[4] e le lingue Sámi[5]. Il presente lavoro costituisce una base fondamentale per concretizzare tale offerta anche per la lingua ladina.

3. Raccolta dei testi e creazione del corpus

I modelli di traduzione automatica neurale vengono idealmente addestrati con grandi quantità di testi, perché è un metodo che insegna a tradurre in modo autonomo e ha quindi bisogno di numerose traduzioni esemplari. I dati di addestramento sono costituiti da coppie di testi paralleli, in cui ogni frase nella lingua di origine è abbinata alla traduzione corrispondente nella lingua di destinazione. Analizzando questi testi paralleli, gli algoritmi assumono le competenze necessarie (grammatica e vocabolario) per la traduzione. Più sono ampi e vari i dati di addestramento, più sono elevate le capacità di questi sistemi.

Questa sezione descrive il metodo utilizzato per costruire un corpus di traduzione automatica per la variante ladina della Val Badia. Abbiamo tratto testi monolingui da un giornale, categorizzato la variante corrispondente e poi applicato un approccio in due fasi che comprende la traduzione dei testi monolingui tramite un sistema di traduzione basata su regole e seguita da una rifinitura.

[3] <https://textshuttle.com/>.

[4] <https://translate.ut.ee/>.

[5] <https://translate.ling.helsinki.fi/ui/sami>.

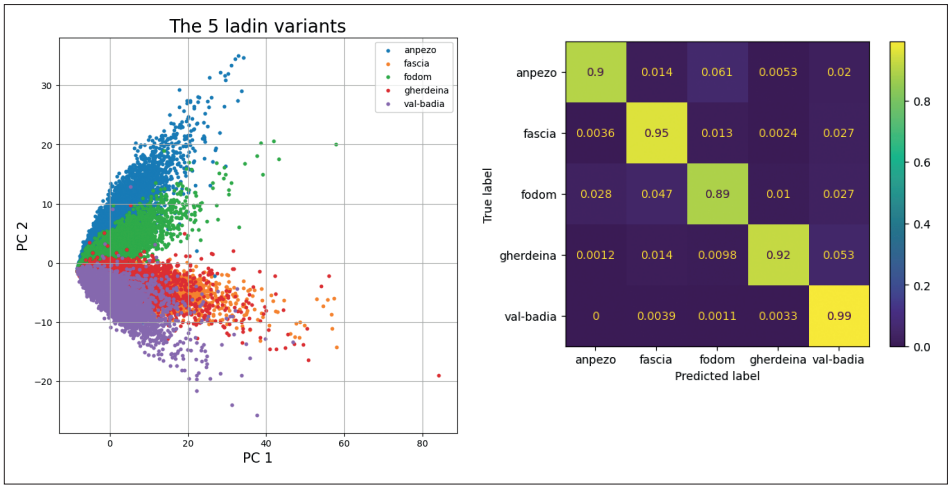

Fig. 2: *Embeddings* ridimensionati in 2D e matrice di confusione del classificatore.

3.1 Classificatore di varianti

Il ladino è suddiviso in cinque varianti principali. Il settimanale "La Usc di Ladins" pubblica articoli in ognuna di queste varianti ed è quindi una fonte importante di testi ladini per quanto riguarda la lingua scritta. Abbiamo tratto la maggior parte dei testi da questa fonte. Poiché le diverse varianti non devono essere confuse, soprattutto quando si traduce in ladino, abbiamo classificato e filtrato i testi prima di procedere all'ulteriore elaborazione.

Per sviluppare un classificatore di varianti per la lingua ladina, abbiamo raccolto estratti di articoli di giornale dal sito web[6] de "La Usc di Ladins", che categorizza i suoi articoli in base alle rispettive vallate e alle corrispondenti varianti linguistiche. Abbiamo raccolto 7.766 estratti di articoli con un totale di 42.745 frasi individuali. Queste frasi sono state poi utilizzate per addestrare un classificatore *XGBoost* (cf. Chen/Guestrin 2016). Abbiamo ottenuto il 94,48% di accuratezza basandoci esclusivamente su 2.500 trigrammi come caratteristiche dei testi *(features)*.

La Figura 2 mostra le caratteristiche ridimensionate in 2D sulla sinistra. Sono riconoscibili chiari raggruppamenti e di conseguenza è possibile una distinzione relativamente accurata delle varianti.

[6] <https://www.lausc.it>.

La parte destra della Figura 2 mostra la matrice di confusione del classificatore sui dati di prova e fornisce una panoramica delle classificazioni del modello in rapporto alle classi effettive per il set dei dati di prova. In generale, il classificatore fornisce prestazioni abbastanza affidabili, con un'alta percentuale di veri positivi per la maggior parte delle classi. Tuttavia, ci sono alcuni errori di classificazione, soprattutto tra Ampezzo, Fassa e Livinallongo, le cui distribuzioni di caratteristiche sembrano sovrapporsi.

La Val Badia si distingue con un tasso di veri positivi alto (0,99), il che significa che il 99% dei testi della Val Badia sono stati identificati. Le percentuali dei falsi negativi sono minimi (*Fascia* 0,39%, *Fodom* 0,11% e *Gherdëina* 0,33%). D'altra parte, l'elevata precisione significa che la percentuale di falsi positivi non è ottimale (*Gherdëina* 5,3%, *Fodom* 2,7%, *Fascia* 2,7%, *Anpezo* 2,0%). Il modello tende quindi a privilegiare "Val Badia" nella sua classificazione, il che è in parte spiegabile in base alla distribuzione sbilanciata dei dati di addestramento. Ma poiché i testi vengono ulteriormente processati, è sufficiente un filtraggio approssimativo che esclude pochi testi d'interesse.[7]

3.2 Dati monolingui

Il giornale ladino "La Usc di Ladins" è archiviato digitalmente dal 2012. Si tratta di un ampio *dataset* di testi monolingui. I testi sono stati estratti e suddivisi in "singole frasi" (per lo più frasi complete, ma anche brevi frasi o singole parole). In totale, abbiamo raccolto 1.937.608 frasi e abbiamo utilizzato il nostro classificatore di varianti per classificare ogni singola frase.

variante	frasi	caratteri
Val Badia	746.704	71.619.515
Gherdëina	491.575	57.704.414
Fascia	407.605	52.504.357
Fodom	146.049	16.615.059
Anpezo	145.674	16.425.301

Tab. 1: Classificazione delle varianti nelle frasi estratte da "La Usc di Ladins".

[7] Nell'applicazione del nostro approccio ad altre varianti, tuttavia, questo aspetto deve essere tenuto presente perché potrebbe essere necessario un aggiustamento per evitare di scartare un numero eccessivo di testi.

```
echo "Chesc é scrit falé." | lt-proc apertium-lld-ita/lld-ita.automorf.bin
^Chesc/*Chesc$ ^é/ester<vbser><pri><p3><sg>/ester<vbser><pri><p3><pl>$
^scrit/scrit<n><m><sg>/scrit<adj><m><sg>/scrí<vblex><pp><m><sg>$
^falé/falé<adv>/falé<vblex><inf>/falé<adj><m><sg>/falé<vblex><pp><m><
sg>$^./.<sent>$
```

Fig. 3: *Apertium* come correttore ortografico.

La Tab. 1 rappresenta i numeri che descrivono la distribuzione delle classificazioni. La terza colonna riporta il numero totale di caratteri. Sottolineiamo che i numeri riportati nella Tab. 1 non riflettono la distribuzione reale dei testi, ma questo numero si basa sulla classificazione del nostro classificatore.

746.704 frasi sono state classificate come "Val Badia".[8] Queste frasi sono state poi ulteriormente analizzate ed elaborate con *Apertium* (cf. Forcada/Tyers 2016). Poiché *Apertium* conosce tutte le possibili flessioni delle parole, è adatto anche come correttore ortografico. Segnala le parole che non riconosce. La Figura 3 illustra un esempio, dove la parola *Chesc*[9] evidenziata in rosso è contrassegnata da un * nell'*output* dell'analisi morfologica per indicare che la parola è sconosciuta al sistema.

Abbiamo utilizzato questa caratteristica per filtrare i dati monolingui, in modo da ottenere una base di dati pulita che faciliti anche le traduzioni basate su regole. Abbiamo scartato le frasi che contengono parole sconosciute (non presenti nel dizionario, e non conosciute da *Apertium*). Una parola sconosciuta può essere:

a) errata (non rispondente alla nuova ortografia introdotta nel 2015),
b) corretta, ma non contenuta nel dizionario,
c) specifica di una variante,
d) illegittima.

[8] In base all'accuratezza del modello di classificazione osservata sui testi di prova, supponiamo che circa 90.000 frasi siano state erroneamente classificate come Val Badia e che circa 10.000 frasi siano sfuggite al modello.

[9] L'ortografia corretta è *Chësc* (-e- con la dieresi).

Nel caso a) le abbiamo aggiunte a un elenco con la correzione del refuso corrispondente, nel caso b) abbiamo aggiunto le parole/sequenze di parole al dizionario di *Apertium* e negli altri due casi abbiamo escluso le parole e quindi le frasi che le contengono. Procedendo in modo iterativo, abbiamo aggiunto sempre più frasi al corpus monolingue, garantendo comunque un certo grado di qualità delle frasi.

In totale, abbiamo definito 9.529 correzioni di errata e aggiunto 1.086 nuove parole al dizionario con le relative traduzioni (la maggior parte erano nomi propri), raccogliendo un totale di 234.236 frasi (ovvero ≈ 31% delle frasi estratte in precedenza). Si potrebbe raccogliere un numero ancora maggiore di testi dedicando ulteriore impegno all'analisi e alla preparazione degli stessi; tra i 472.468 testi non utilizzati, ce ne sono ≈ 110.000 che contengono solo una parola/tipologia sconosciuta.

3.3 Costruzione del corpus bilingue

Per la costruzione del corpus bilingue seguiamo un approccio in due fasi. In una prima fase, traduciamo i testi monolingui ladini precedentemente filtrati dall'italiano utilizzando *Apertium* e in una seconda fase li rifiniamo ulteriormente.

Come è noto, le traduzioni generate da sistemi basati su regole non sono di alta qualità, soprattutto per quanto riguarda l'aspetto della fluidità (in Pourdamghani 2019 viene definita *Translationese*). Inoltre, il problema dell'ambiguità si presenta spesso e rappresenta la principale limitazione e sfida di questo approccio. Nella traduzione basata su regole, spesso non si tiene conto del contesto delle parole e la scelta della traduzione nel caso di parole ambigue non può essere sempre definita da regole. Abbiamo implementato il sistema in modo che scelga sempre il primo suggerimento, che ovviamente spesso è quello sbagliato e falsifica il significato dei testi. Per ovviare a questo problema e per migliorare ulteriormente il sistema di traduzione basato su regole, abbiamo estratto dai testi gli *n*-grammi di parole più comuni e abbiamo aggiunto le loro traduzioni corrispondenti come regole. In totale, abbiamo aggiunto di 981 2-, 3-, 4- e 5-grammi di parole. Il sistema *Apertium* risultante ha ottenuto 15,92 punti BLEU sul nostro set di prova.[10]

[10] Su Collodi 2017 questo sistema raggiunge +9,24 BLEU rispetto allo stato del sistema in Frontull/Hell 2022.

In una seconda fase, affiniamo ulteriormente le traduzioni sintetiche elaborandole attraverso l'*API* di *DeepL*. Mediante questo processo si migliora il risultato traducendo le frasi italiane, generate dal sistema di traduzione *Apertium*, dall'italiano all'inglese per tornare all'italiano. Si tratta di una forma di correzione automatica delle frasi. La Tab. 2 mostra alcuni esempi di risultati di questo processo di affinamento. La prima riga *(X.0)* nei blocchi di questa tabella rappresenta la frase ladina di partenza. La seconda riga (X.1) è la traduzione generata da *Apertium* che presenta diversi errori di traduzione e imprecisioni grammaticali. Sottoponendo queste traduzioni all'*API* di *DeepL* e quindi riformulandole, possiamo osservare un miglioramento significativo della qualità della traduzione (X.2). Tuttavia, funziona solo se le traduzioni preliminari non distorcono il significato.

1.0	*I un pumpé dui dis alalungia za. 15.000 litri d'ega al menüt.*
1.1	Abbiamo pompare due giorni per circa 15.000 litri di'acqua a minuto.
1.2	Per due giorni abbiamo pompato circa 15.000 litri d'acqua al minuto.
2.0	*Ara á adoré le dotur deache ara â borjú.*
2.1	Ha adorato il dottore perché ella avevo febbre.
2.2	Adorava il dottore perché aveva la febbre.
3.0	*T'es vigni de te nüsc pinsiers y te restaras dagnora te nüsc cörs.*
3.1	Te'sei ogni giorno in nostri pensieri e te resterai sempre in nostri cuori.
3.2	Siete nei nostri pensieri ogni giorno e rimarrete sempre nei nostri cuori.
4.0	*Tl medem momënt dess düc ponsé sura por tan dî y por ci ch'ai adora chësc meso de comunicaziun.*
4.1	Nello stesso momento davo tutti pensato sopra per quanto tempo e per che che presto questo mezzo di comunicazione
4.2	Allo stesso tempo pensavo a quanto a lungo e per quanto tempo questo mezzo di comunicazione sarebbe stato in grado di funzionare.

Tab. 2: Rifinitura delle traduzioni attraverso l'*API* di *DeepL*.

Il blocco 1 mostra un esempio in cui questa correzione automatica funziona bene. La traduzione basata su regole (1.1) è comprensibile e conserva il significato, ma presenta difetti grammaticali. Questi esempi possono essere migliorati in modo significativo nel processo di affinamento, come si può vedere dalla correzione generata automaticamente tramite *DeepL* (1.2).

Il blocco 2 presenta un esempio di frase in cui la parola *adoré* viene tradotta in modo errato a causa della sua ambiguità (2.1). A seconda del contesto, *adoré* in

ladino può significare "usare", "bisogno" o "adorazione". In questo caso, il significato inteso è "bisogno", ma il sistema lo ha tradotto in "adorare", alterando così il significato della frase. Questo errore persiste anche durante il processo di affinamento (2.2).

Il blocco 3 presenta un esempio in cui la traduzione viene distorta a causa della traduzione ausiliaria inglese. In ladino, *T'es* significa "Tu sei", ma viene tradotto come "Siete", perché l'inglese manca di questa distinzione (in entrambi i casi si dice *You are*). Questo esempio suggerisce di riflettere su quale sia la lingua ausiliare più adatta per la correzione.

Il blocco 4 presenta un esempio in cui la traduzione basata su regole non funziona bene e in cui la correzione automatica è inefficace. Tali casi possono deteriorare significativamente la qualità complessiva e dovrebbero essere esclusi (come dimostrano i nostri esperimenti, di cui si parlerà più avanti).

Oltre alle 234.236 frasi tratte dal giornale, abbiamo raccolto 3.296 frasi dalla pagina web di *Trail*[11] e ≈ 1.000 da altre fonti e abbiamo sintetizzato le loro traduzioni come appena illustrato. Abbiamo anche raccolto ≈ 4.500 frasi parallele da diverse pubblicazioni bilingui, ad esempio Collodi 2017, e le abbiamo aggiunte al corpus. Dal dizionario abbiamo estratto 19.043 frasi brevi e parzialmente incomplete e 27.330 locuzioni. In totale, abbiamo accumulato 296.956 testi paralleli (di cui circa il 17% sono traduzioni autentiche).

[11] <https://www.rainews.it/tgr/trail>.

4. NMT con *transformers*

Abbiamo utilizzato l'architettura più avanzata per la traduzione automatica neurale chiamata *transformer* (cf. Vaswani et al. 2017). I *transformers* hanno guadagnato popolarità nel campo della traduzione automatica neurale perché sono in grado di interpretare in modo efficiente le dipendenze a lunga distanza nei testi, rendendo le traduzioni più precise e fluide. Inoltre, il loro processo di addestramento è parallelizzabile, ciò consente di addestrarli in maniera efficiente su grandi quantità di dati.

BPE	*le dotur é gnü cherdé te ospedal por n'emergënza*
500	le d_ ot_ ur é gnü cher_ dé te os_ pe_ dal por n'_ e_ mer_ g_ ën_ za.
1k	le d_ ot_ ur é gnü cherdé te os_ pe_ dal por n'_ e_ mer_ g_ ën_ za.
2k	le d_ ot_ ur é gnü cherdé te os_ pe_ dal por n'_ e_ mer_ g_ ënza.
4k	le dotur é gnü cherdé te ospe_ dal por n'_ e_ mer_ g_ ënza.
8k	le dotur é gnü cherdé te ospedal por n'_ emerg_ ënza.
16k	le dotur é gnü cherdé te ospedal por n'_ emergënza.

Tab. 3: Variazioni della tokenizzazione con un diverso numero di operazioni di fusione BPE.

Abbiamo addestrato *Transformer* con vari *subset* del corpus testuale da zero e anche allenato modelli di traduzione esistenti tra italiano e spagnolo. In questa sezione discutiamo i due approcci e i risultati ottenuti.

4.1 Formazione da zero

Seguendo le indicazioni[12] riportate in Sennrich/Zhang 2019 e Araabi/Monz 2020 abbiamo addestrato *Transformer* su una *GPU Nvidia RTX A2000 Laptop* con 8 GB utilizzando il *toolkit OpenNMT-tf* (cf. Klein et al. 2017). Abbiamo interrotto l'addestramento del sistema, quando i tentativi di miglioramento erano inferiori a 0,2 punti BLEU. Di seguito elenchiamo i parametri più importanti che abbiamo utilizzato.

[12] Non tutte le impostazioni sono compatibili con la nostra GPU.

4.1.1 Tokenizzazione

La tokenizzazione svolge un ruolo cruciale nella NMT, soprattutto in scenari a basse risorse (cf. Domingo et al. 2019). Prima di elaborare i dati paralleli, i testi devono essere tokenizzati, cioè suddivisi in singoli *tokens*. A seconda del metodo di tokenizzazione, un *token* può essere una sequenza di parole, una singola parola, una sottoparola o un singolo carattere. Nei nostri esperimenti abbiamo utilizzato l'algoritmo del *byte pair encoding* (BPE), descritto in Sennrich/Haddow/Birch 2016b, che è un algoritmo di tokenizzazione consolidato.

L'idea principale di BPE è quella di unire iterativamente le sequenze di caratteri più frequenti in un corpus, fino a raggiungere una determinata dimensione del vocabolario. Con questo procedimento viene creato un vocabolario di *token* che vengono poi usati per codificare le parole nel testo. Il BPE si è dimostrato efficace nell'individuare le informazioni morfologiche e compositive delle parole, particolarmente utili per le lingue con flessioni complesse. Consente ai modelli neurali di gestire parole rare e variazioni morfologiche suddividendole in unità di sottoparole significative.

Più operazioni di fusione vengono eseguite, più *tokens* diversi si accumulano (dimensione del vocabolario). Con un vocabolario più ampio, tuttavia, si rischia di perdere la versatilità dello strumento. La Tab. 3 dà un idea di come il numero di operazioni di fusione influisca sul modo in cui i testi vengono tokenizzati.

Abbiamo addestrato *Transformer* su vocabolari di diverse dimensioni e ne abbiamo analizzato le prestazioni. La Tab. 4 elenca i testi prodotti dai modelli addestrati con una tokenizzazione diversa. È evidente che, con vocabolari più ampi, singole parole possono essere completamente sfasate nella traduzione, mentre con vocabolari ridotti le traduzioni contengono errori di tipo più fine che possono essere spiegati dalla composizione di *tokens* più brevi. Questo si può osservare anche nelle traduzioni generate. Sulla base di questo esperimento, abbiamo deciso di utilizzare 1k operazioni di fusione nei nostri esperimenti, ottenendo così un vocabolario di dimensioni pari a ≈ 1.200 *tokens*. Abbiamo utilizzato lo strumento *subword-nmt*[13] per applicare l'algoritmo BPE e per tokenizzare i dati.

[13] <https://github.com/rsennrich/subword-nmt>.

source	*Diversi fattori influenzano la qualità delle traduzioni prodotte, rendendo complessa l'identificazione della configurazione ottimale.*
500	Deplü faturs influenzëia la cualité dles traduziuns produtes, fajon complessëia l'identificaziun dla configuraziun otimala.
1k	Deplü faturs influenzëia la cualité dles traduziuns produtes, fajon comples l'identificaziun dla configuraziun otiva.
2k	Deplü faturs influenzëia la cualité dles traduziuns produtes, meton adöm l'identificaziun dla configuraziun otimala.
4k	Deplü influenzëia la cualité dles traduziuns produtes, fajon lapró l'identificaziun dl configuraziun otica.
8k	Deplü faturs á influenzé la cualité dles traduziuns de produt, y capí l'identificaziun dla configuraziun dër bun.
16k	An pó influenzé la cualité dles traduziuns produtes, ti fá comples l'identificaziun dla conzeziun dla atuaziun.
riferimento	*La cualité dles traduziuns depënn da deplü faturs, al é compliché da ciafé la configuraziun otimala.*

Tab. 4: Effetto del numero di operazioni di fusione BPE sulle traduzioni generate.

4.1.2 Configurazione del modello

Abbiamo configurato il *transformer* con quattro *layer*, ciascuno composto da 512 *hidden units* e due *attention heads*. La rete neurale *feedforward* (FFN) ha una dimensione interna di 1.024 unità. Abbiamo impostato una lunghezza massima di 120 *tokens* e una dimensione del *batch* di 4.096 *tokens*.

Questa impostazione dei parametri stabilisce un equilibrio tra la complessità del modello e l'efficienza computazionale. Con più *layer*, *attention heads*, una dimensione *feed forward* o del *batch* maggiore, il modello potrebbe essere in grado di identificare strutture più complesse ed essere addestrato in modo più affidabile, ma ciò è associato a costi computazionali elevati e a tempi di addestramento più lunghi. Inoltre, è stato dimostrato che i modelli più compatti funzionano meglio in contesti con poche risorse (cf. Sennrich/Zhang 2019).

4.1.3 *Beam search*

I modelli di traduzione basati su statistiche, così come anche i modelli neurali, producono traduzioni pronosticando in modo iterativo il *token* successivo più probabile. Invece di scegliere l'opzione più probabile (decodifica *greedy*), il *beam search* permette di esplorare diverse possibilità di traduzione e quindi di trovare una traduzione più coerente al contesto.

In ogni fase di decodifica, il *beam search* considera i top-*k* candidati, dove *k* è la dimensione del *beam*, un parametro definito dall'utente. L'algoritmo genera un elenco di possibili *tokens* successivi per ogni candidato e assegna un punteggio in base alla loro probabilità. I *k* candidati con il punteggio più alto vengono selezionati e la procedura viene ripetuta fino a quando viene raggiunta una condizione di terminazione (fine della sequenza o lunghezza massima della frase). Abbiamo scelto una dimensione del *beam* di 3.

L'esperienza ha dimostrato che un *beam* più grande, specialmente in contesti a basse risorse, non apporta miglioramenti. Esiste il fenomeno per cui la qualità della traduzione addirittura diminuisce con un *beam* più grande (cf. Yang/Huang/Ma 2018).

4.1.4 *Dropout*

Il *dropout* è una tecnica di regolarizzazione che esclude in modo casuale una parte delle unità di *input* durante l'addestramento per evitare un sovraadattamento *(overfitting)* e migliorare la versatilità del modello. Abbiamo scelto un *dropout* di 0,1, il che significa che in ogni iterazione il 10% dei *tokens* viene nascosto in modo casuale. Questo costringe il modello ad apprendere caratteristiche più resistenti e indipendenti, riducendo allo stesso tempo la sua dipendenza da neuroni specifici. Non applichiamo una regolarizzazione aggressiva del *dropout* come suggerito in Sennrich/Zhang 2019.

4.2 *Transfer learning*

Il concetto del *transfer learning* "apprendimento per trasferimento", presentato originariamente in Zoph et al. 2016, si basa sull'idea di utilizzare conoscenze e competenze esistenti per finalità simili e di adattarle a un nuovo obiettivo. Nel contesto della traduzione automatica, ad esempio, modelli di traduzione automatica esistenti possono essere ulteriormente addestrati ad una nuova coppia di lingue in un processo di *fine tuning*. Questo permette di beneficiare delle competenze linguistiche già acquisite, ed è stato dimostrato che questo è un metodo efficace, anche in un'applicazione triviale come quella da noi perseguita e descritta in Kocmi/Bojar 2018. L'ottimizzazione dei modelli pre-addestrati è quindi un approccio comune nel campo della NMT. Questo approccio può ridurre significativamente i tempi di addestramento e i requisiti di dati e spesso porta a una migliore qualità della traduzione, particolarmente preziosa in scenari a basse risorse.

Abbiamo addestrato modelli di traduzione che fanno parte del progetto *Opus-MT* (cf. Tiedemann/Thottingal 2020). Sono disponibili su *Hugging Face*[14] e possono essere utilizzati tramite la *library Transformers* (cf. Wolf et al. 2020). Abbiamo scelto lo spagnolo come lingua ausiliaria, perché ha una struttura simile a quella del ladino. Abbiamo addestrato i modelli *Helsinki-NLP/opus-mt-it-es* per tradurre dall'italiano al ladino e *Helsinki-NLP/opus-mt-es-it* per tradurre dal ladino all'italiano.

4.3 Risultati

In questo esperimento, abbiamo valutato le prestazioni del nostro sistema di traduzione automatica su una pubblicazione piuttosto sofisticata e culturalmente significativa, "Alto Adige: un'Europa in piccolo – I 50 anni del Secondo Statuto di autonomia" (Kager 2022), da cui abbiamo estratto 900 frasi parallele. Questa pubblicazione commemorativa in occasione del 50° anniversario del Secondo Statuto di autonomia dell'Alto Adige rappresenta una combinazione di complesse osservazioni storiche, regionali e culturali. Sottoponendo il nostro sistema a questa pubblicazione, abbiamo voluto valutare la capacità di tradurre adeguatamente argomenti complessi. La Tab. 5 elenca i risultati dei diversi esperimenti condotti. Riportiamo i punti BLEU ottenuti dai rispettivi modelli. La traduzione basata su regole è stata ottimizzata in modo specifico per la direzione di traduzione dal

[14] <https://huggingface.co/>.

corpus	frasi	approccio	It → La	La → It
–	–	basato su regole	4,41	15,92
dati originari	47.091	da zero	2,20	2,27
		transfer learning	7,62	14,34
+ testi monolingui ladini	281.327	da zero	16,13	13,96
(con dati originari)		*transfer learning*	17,61	13,67
+ correzione	281.327	da zero	20,61	22,18
automatica		*statistical*	18,31	18,79
		transfer learning	19,96	21,36
+ filtro RTT ≥ 0,4	237.129	da zero	**21,11**	**22,43**
		transfer learning	19,50	21,42
RTT ≥ 0,6	184.708	da zero	20,63	20,29
		transfer learning	18,73	21,41

Tab. 5: Punteggi BLEU di diversi modelli di traduzione valutati in Kager 2022.

ladino all'italiano (15,92 BLEU), poiché questa direzione era cruciale per la traduzione preliminare. La direzione opposta non è stata affrontata (solo 4,41 BLEU). Utilizzando solo dati di addestramento autentici, i nostri esperimenti dimostrano che l'approccio del *transfer learning* è chiaramente superiore all'addestramento dei modelli da zero *(from-scratch)* in entrambe le direzioni di traduzione, come dimostrano i punteggi BLEU 7,62 contro 2,20 (It → La) e 14,34 contro 2,27 (La → It). Questa evidenza empirica sottolinea l'efficienza dell'approccio del *transfer learning*, che con uno sforzo minimo è quasi alla pari con la traduzione basata su regole nella direzione La → It (e anche nella direzione opposta).

Sulla base dei testi processati con *DeepL*, sono stati raggiunti miglioramenti significativi nella qualità della traduzione, soprattutto nella direzione di traduzione La → It. Osserviamo un aumento di +8,22 BLEU per l'approccio *from-scratch* e di +7,69 BLEU per l'approccio *transfer learning*. Per un confronto, abbiamo riportato anche le prestazioni dell'approccio statistico per questi dati.

Inoltre, abbiamo esaminato due criteri di filtraggio basati sulla cosiddetta *round-trip translation* (RTT), ovvero traduzione di andata e ritorno, come presentato in Imankulova/Sato/Komachi 2020. Abbiamo escluso le traduzioni sintetizzate la cui traduzione di ritorno aveva un punteggio BLEU inferiore alle soglie di 0,4 (escludendo circa 45.000 frasi) e 0,6 (escludendo circa 97.000 frasi). Questo concetto ha portato ai migliori risultati e sottolinea l'importanza del controllo di qualità. Inoltre suggerisce anche l'esistenza di un inquinamento dei dati di addestramento.

Nell'approccio d'addestramento da zero sono stati raggiunti 21,11 BLEU nella direzione It → La, e 22,43 nella direzione La → It. L'approccio *transfer learning* non presenta differenze significative in riferimento a questi criteri.

5. *Tool*

Un aumento del numero di frasi non sempre porta a un miglioramento della qualità di traduzione (cf. Imankulova/Sato/Komachi 2017). Soprattutto quando si tratta di dati sintetici, può essere di grande beneficio perfezionare i dati, se possibile. Abbiamo implementato un'applicazione web che consente di modificare e correggere i testi. Questo permette agli specialisti di migliorare ulteriormente il traduttore. L'applicazione web è basata su *Python* e realizzata con il *framework Flask*.[15] Nella versione più recente, utilizza *MongoDB*[16] come sistema di gestione del *database*.

Abbiamo anche integrato *Apertium* come correttore ortografico. Questo permette di evidenziare le parole che non sono contenute nel dizionario e di fornire suggerimenti di correzione. La Fig. 4 mostra una schermata esemplare che illustra questa funzionalità su un estratto di un articolo di cronaca.

È inoltre possibile esaminare i dati paralleli e cercare specifici errori di traduzione, come ad esempio traduzioni errate di parole ambigue, come abbiamo già visto per *adoré*. La Fig. 5 mostra una schermata dell'applicazione che riporta tutti i testi del corpus in cui *adoré* ricorre nel testo ladino e "adorare" in quello italiano. In entrambi gli esempi raffigurati la traduzione con "adorare" è sbagliata.

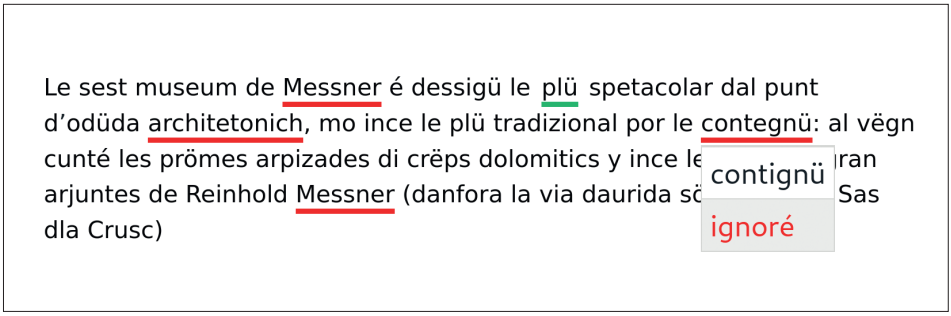

Fig. 4: Correzione e modifica dei testi nel *tool*.

[15] <https://flask.palletsprojects.com/en/2.3.x/>.

[16] <https://www.mongodb.com/>.

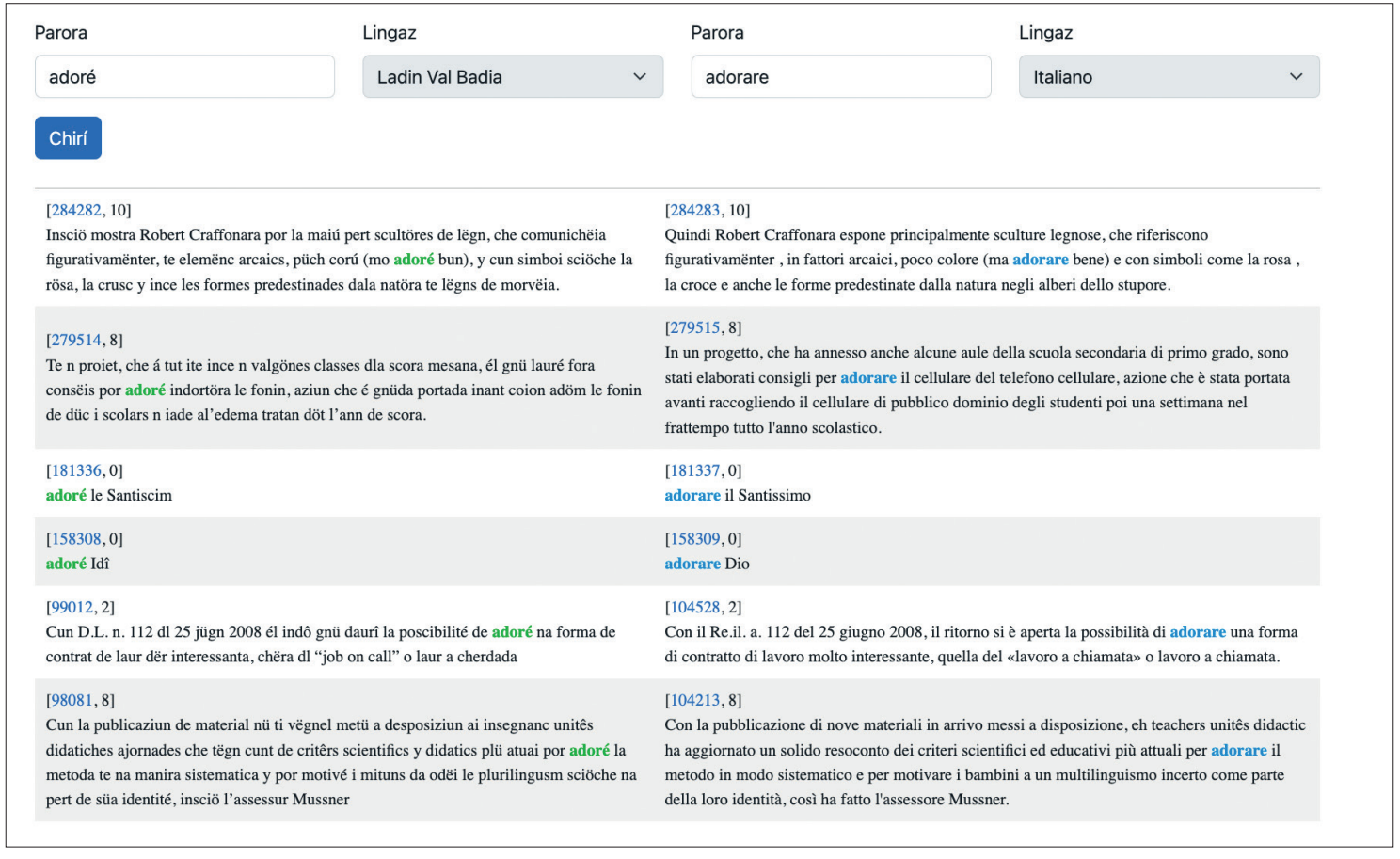

Fig. 5: Ricerca di errori di traduzione e modifica dei dati paralleli.

Questi errori possono essere corretti direttamente. Grazie a questi interventi manuali, non solo vengono corretti gli errori, ma vengono creati ulteriori dati paralleli convalidati. Riteniamo quindi che questi interventi abbiano un grande potenziale per ulteriori miglioramenti del sistema.

6. Conclusione

Abbiamo presentato un approccio alla traduzione automatica neurale per la lingua ladina. In particolare, abbiamo applicato la tecnica della *back-translation* per sintetizzare dati complementari che hanno consentito l'addestramento di *transformer*, sia da zero che sulla base di modelli esistenti. Entrambi gli approcci sono stati valutati su testi piuttosto impegnativi tratti dalla pubblicazione "Alto Adige: un'Europa in piccolo – I 50 anni del Secondo Statuto di autonomia" (Kager 2022). Sono stati ottenuti risultati soddisfacenti, significativamente migliori di quelli ottenuti con gli approcci tradizionali precedentemente studiati. La tecnica della *back-translation* si è dimostrata efficace. Nella traduzione dall'italiano al ladino, abbiamo ottenuto un punteggio BLEU di 21,11 e nella direzione opposta, dal ladino all'italiano, abbiamo raggiunto un punteggio BLEU di 22,43. Abbiamo reso accessibili i modelli sviluppati attraverso una applicazione web. Dalla schermata Figura 6 si evince che il sistema dà un buon risultato di traduzione, ma non riesce, in questo caso, a individuare il genere corretto. Infatti, la traduzione del sostantivo "insegnante" (s.m. e s.f.) in ladino dà impropriamente come risultato la forma *la maestra* "la maestra" (s.f.). Ecco perchè ci siamo ripromessi di sviluppare una piattaforma che consenta di correggere i testi e quindi migliorare i sistemi addestrati su di essi.

Gli approcci esaminati hanno suscitato ulteriori quesiti di ricerca. Uno di questi è quale sia la lingua più adatta per la fase di rifinitura delle traduzioni sintetizzate con il sistema di traduzione basato su regole, per la quale abbiamo scelto l'inglese. L'uso di una lingua che non appartiene alla stessa famiglia può essere vantaggioso perché certi errori non possono essere trasferiti per inerzia, ma può anche essere problematico perché informazioni importanti potrebbero essere smarrite.

Una questione simile è quella relativa a quale lingua ausiliaria sia la più adatta per l'approccio di *transfer learning*. È già stato dimostrato che le lingue affini sono in genere più efficaci (cf. Goyal/Kumar/Sharma 2020). Nel caso del ladino, si potrebbero prendere in considerazione, accanto all'italiano, diverse lingue romanze ad alte risorse, come ad esempio il francese, il portoghese o il rumeno.

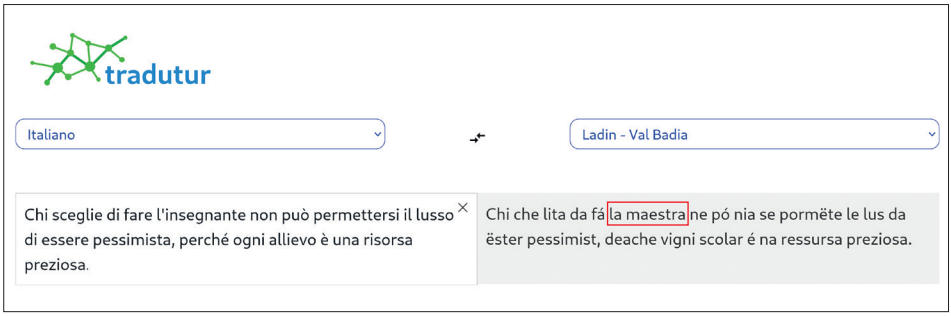

Fig. 6: Schermata del *tradutur* che mostra la traduzione ladina generata di una citazione di Luca Serianni.

Gli ultimi sviluppi nella ricerca della traduzione automatica per lingue a basse risorse si stanno muovendo nella direzione di modelli multilingui, poiché questo permette di usufruire contemporaneamente di diverse lingue correlate ad alte risorse (cf. Lakew 2019, Neubig/Hu 2018). L'uso di modelli multilingui consentirebbe inoltre di sviluppare un modello NMT unificato in grado di gestire tutte le varianti della lingua ladina. Pertanto, intendiamo studiare anche questo approccio in futuro.

Sviluppando modelli multilingui, anche la valutazione diventa più interessante. Il *benchmark* di valutazione Flores-101 (cf. Goyal et al. 2022) comprende testi paralleli in oltre 200 lingue. La nostra intenzione è di estendere questa collezione includendo le traduzioni ladine (per le singoli varianti), in modo da poter valutare i nostri sistemi in un contesto multilingue.

Infine, l'integrazione dei *Large Language Models* (LLM), come i modelli GPT,[17] è una direzione di ricerca promettente. Anche se non sono ancora all'altezza di sistemi specializzati (cf. Zhang/Haddow/Birch 2023), potrebbero comunque essere utili per lingue a basse risorse. Tali modelli potrebbero essere utilizzati per produrre traduzioni sintetiche preliminari qualitativamente migliori di quelle generate attraverso un sistema basato su regole. Nel *prompt*, ad esempio, si potrebbero specificare i diversi significati di ogni parola in una frase e lasciare scegliere al LLM quello più adatto.

[17] <https://platform.openai.com/docs/guides/gpt>.

7. Limitazioni

La metodologia in due fasi, che prevede l'utilizzo di un sistema di traduzione basato su regole per la traduzione preliminare e il successivo perfezionamento tramite l'*API* di *DeepL*, si è dimostrata una strategia efficace. Tuttavia, abbiamo anche osservato che la traduzione basata su regole introduce diversi errori, in particolare dovuti ad ambiguità, che persistono nella fase di perfezionamento (cf. Tab. 2 e Fig. 6).

Invece di affidarsi alla traduzione basata su regole, una strategia migliore potrebbe essere l'utilizzo di un modello addestrato esclusivamente con dati autentici in un approccio di *transfer learning*. Nei nostri esperimenti abbiamo ottenuto con un modello di questo tipo prestazioni paragonabili a quelle del sistema basato su regole. Per questo motivo sarebbe interessante valutare questa soluzione alternativa.

Abbiamo tradotto solo testi monolingui ladini. Pertanto, la maggior parte dei testi italiani del corpus era di bassa qualità (perché sintetizzati). Soprattutto nella direzione di traduzione La → It, questo può avere un impatto negativo sulla qualità della traduzione. Per questo motivo, il corpus dovrebbe in ogni caso essere arricchito anche con testi monolingui italiani, al fine di migliorare ulteriormente questa direzione di traduzione.

Inoltre, riteniamo che vi sia un potenziale di ulteriore miglioramento con un approccio iterativo della *back-translation* (cf. Hoang et al. 2018), come dimostrato in un recente studio sulla traduzione automatica per il livoniano (cf. Rikters et al. 2022).

8. Abbreviazioni

API	Abstract Programming Interface
BLEU	Bilingual Evaluation Understudy
BPE	Byte Pair Encoding
FFN	Feedforward
GB	Gigabyte
GPT	Generative Pre-trained Transformer
GPU	Graphics Processing Unit
LLM	Large Language Models
NLLB	No Language Left Behind
NLP	Natural Language Processing
NMT	Neural Machine Translation
RTT	Round-Trip Translation

9. Bibliografia

Araabi, Ali/Monz, Christof: *Optimizing Transformer for Low-Resource Neural Machine Translation*, in: Scott, Donia/Bel, Nuria/Zong, Chengqing (eds.), Proceedings of the 28th international conference on computational linguistics, Barcelona 2020, 3429–3435; <https://aclanthology.org/2020.coling-main.304.pdf>.

Bapna, Ankur et al.: *Building Machine Translation Systems for the Next Thousand Languages*; <https://doi.org/10.48550/arXiv.2205.03983>, [06/07/2022].

Bapna, Ankur/Firat, Orhan: *Simple, Scalable Adaptation for Neural Machine Translation*, in: Inui, Kentaro et al. (eds.), Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong 2019, 1538–1548.

Caswell, Isaac/Chelba, Ciprian/Grangier, David: *Tagged Back-Translation*, in: Bojar, Ondřej et al. (eds.), Proceedings of the Fourth Conference on Machine Translation (volume 1: Research papers), Florence 2019, 53–63; <https://doi.org/10.48550/arXiv.1906.06442>, [15/06/2019].

Chen, Tianqi/Guestrin, Carlos: *XGBoost: A Scalable Tree Boosting System*, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York 2016, 785–794; <https://doi.org/10.1145/2939672.2939785>, [13/08/2016].

Collodi, Carlo: *Les aventöres de Pinocchio*, San Martin de Tor 2017.

Dabre, Raj/Fujita, Atsushi/Chu, Chenhui: *Exploiting Multilingualism through Multistage Fine-Tuning for Low-Resource Neural Machine Translation*, in: Inui, Kentaro et al. (eds.), Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong 2019, 1410–1416; <https://aclanthology.org/D19-1146>.

Domingo, Miguel et al.: *How Much Does Tokenization Affect Neural Machine Translation?*, in: Gelbukh, Alexander (ed.), Computational Linguistics and Intelligent Text Processing, Cham 2023, 545–554; <https://doi.org/10.48550/arXiv.1812.08621>, [11/06/2019].

Dou, Zi-Yi/Anastasopoulos, Antonios/Neubig, Graham: *Dynamic Data Selection and Weighting for Iterative Back-Translation*, in: Webber, Bonnie et al. (eds.), Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp), 2020, 5894–5904; <https://doi.org/10.48550/arXiv.2004.03672>, [07/10/2020].

Erk, Katrin/Smith, Noah A. (eds.): *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin 2016.

Fadaee, Marzieh/Bisazza, Arianna/Monz, Christof: *Data Augmentation for Low-Resource Neural Machine Translation*, in: Barzilay, Regina/Kan, Min-Yen (eds.), Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Vancouver 2017, 567–573; <https://aclanthology.org/P17-2090.pdf>.

Forcada, Mikel L./Tyers, Francis M.: *Apertium: A free/open source platform for machine translation and basic language technology*, in: Proceedings of the 19th annual conference of the european association for machine translation: Projects/products, Riga 2016, 380; <https://aclanthology.org/2016.eamt-2.4.pdf>.

Frontull, Samuel/Hell, Tobias: *Arbeitsbericht: Maschinelle Übersetzung für das Gadertalische*, in: "Ladinia", XLVI, 2022, 203–232.

Goyal, Naman et al.: *The Flores-101 Evaluation Benchmark for Low-Resource and Multilingual Machine Translation*, in: "Transactions of the Association for Computational Linguistics", 10, 2022, 522–538.

Goyal, Vikrant/Kumar, Sourav/Sharma, Dipti Misra: *Efficient Neural Machine Translation for Low-Resource Languages via Exploiting Related Languages*, in: Rijhwani, Shruti et al. (eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, s.l. 2020, 162–168; <https://aclanthology.org/2020.acl-srw.22.pdf>.

Gu, Jiatao et al.: *Universal Neural Machine Translation for Extremely Low Resource Languages*, in: Walker, Marilyn/Ji, Heng/Stent, Amanda (eds.), Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans 2018, 344–354; <https://aclanthology.org/N18-1032.pdf>.

Haddow, Barry et al.: *Survey of Low-Resource Machine Translation*, in: "Computational Linguistics", 48, 2022, 673–732.

Hoang, Vu Cong Duy et al.: *Iterative Back-Translation for Neural Machine Translation*, in: Birch, Alexandra et al. (eds.), Proceedings of the 2nd Workshop on Neural Machine Translation and Generation, Melbourne 2018, 18–24; <https://aclanthology.org/W18-2703.pdf>.

Imankulova, Aizhan/Sato, Takayuki/Komachi, Mamoru: *Improving Low-Resource Neural Machine Translation with Filtered Pseudo-Parallel Corpus*, in: Nakazawa, Toshiaki/Goto, Isao (eds.), Proceedings of the 4th Workshop on Asian Translation (WAT2017), Taipei 2017, 70–78; <https://aclanthology.org/W17-5704.pdf>.

Imankulova, Aizhan/Sato, Takayuki/Komachi, Mamoru: *Filtered Pseudo-parallel Corpus Improves Low-resource Neural Machine Translation*, in: "ACM Transactions on Asian and Low-Resource Language Information Processing", 19/2, 2020, 1–16; <https://doi.org/10.1145/3341726>.

Kager, Thomas (ed.): *Alto Adige: un'Europa in piccolo – I 50 anni del Secondo Statuto di autonomia*, Bolzano 2022.

Klein, Guillaume et al.: *OpenNMT: Open-Source Toolkit for Neural Machine Translation*, in: Bansal, Mohit/Ji, Heng (eds.), Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics-System Demonstrations, Vancouver 2017, 67–72; <https://aclanthology.org/P17-4012.pdf>.

Kocmi, Tom/Bojar, Ondřej: *Trivial Transfer Learning for Low-Resource Neural Machine Translation*, in: Bojar, Ondřej et al. (eds.), Proceedings of the Third Conference on Machine Translation: Research Papers, Brussels 2018, 244–252; <https://aclanthology.org/W18-6325.pdf>.

Korhonen, Anna/Traum, David/Màrquez, Lluís (eds.): *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence 2019.

Lakew, Surafel M. et al.: *Adapting Multilingual Neural Machine Translation to Unseen Languages*, <https://arxiv.org/pdf/1910.13998.pdf>, [30/10/2019].

Moling, Sara et al.: *Dizionario italiano–ladino Val Badia / Dizionar ladin Val Badia–talian*, San Martin de Tor 2016.

Neubig, Graham/Hu, Junjie: *Rapid Adaptation of Neural Machine Translation to New Languages*, in: Riloff, Ellen et al. (eds.), Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels 2018, 875–880; <https://aclanthology.org/D18-1103.pdf>.

NLLB Team et al.: *No Language Left Behind: Scaling Human-Centered Machine Translation*, Berkeley 2022; <https://arxiv.org/pdf/2207.04672>, [11/07/2022].

Popel, Martin/Bojar, Ondřej: *Training Tips for the Transformer Model*, in: "The Prague Bulletin of Mathematical Linguistics", 110, 2018, 43–70.

Pourdamghani, Nima et al.: *Translating Translationese: A Two-Step Approach to Unsupervised Machine Translation*, in: Korhonen/Traum/Màrquez 2019, op. cit., 3057–3062.

Ranathunga, Surangika et al.: *Neural Machine Translation for Low-Resource Languages: A Survey*, in: "ACM Computing Surveys", 55/11, 2023, 1–37.

Rikters, Matīss et al.: *Machine Translation for Livonian: Catering to 20 Speakers*, in: Muresan, Smaranda/Nakov, Preslav/Villavicencio, Aline (eds.), Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Dublin 2022, 508–514; <https://aclanthology.org/2022.acl-short.55.pdf>.

Sennrich, Rico/Zhang, Biao: *Revisiting Low-Resource Neural Machine Translation: A Case Study*, in: Korhonen/Traum/Màrquez 2019, op. cit., 211–221.

Sennrich, Rico/Haddow, Barry/Birch, Alexandra: *Improving Neural Machine Translation Models with Monolingual Data*, in: Erk/Smith 2016a, op. cit., 86–96.

Sennrich, Rico/Haddow, Barry/Birch, Alexandra: *Neural Machine Translation of Rare Words with Subword Units*, in: Erk/Smith 2016b, op. cit., 1715–1725.

Shi, Shumin et al.: *Low-resource Neural Machine Translation: Methods and Trends*, in: "ACM Transactions on Asian and Low-Resource Language Information Processing", 21/5, 2022, 1–22.

Tiedemann, Jörg/Thottingal, Santhosh: *OPUS-MT – Building open translation services for the World*, in: Martins, André et al. (eds.), Proceedings of the 22nd Annual Conference of the European Association for Machine Translation, Lisboa 2020, 479–480; <https://aclanthology.org/2020.eamt-1.61.pdf>.

van der Kreeft, Peggy et al.: *GoURMET – Machine Translation for Low-Resourced Languages*, in: He Moniz, Helena et al. (eds.), Proceedings of the 23rd Annual Conference of the European Association for Machine Translation, Ghent 2022, 339–340; <https://aclanthology.org/2022.eamt-1.59.pdf>.

Vaswani, Ashish et al.: *Attention Is All You Need*, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach 2017, 6000–6010; <https://doi.org/10.48550/arXiv.1706.03762>, [02/08/2023].

Wang, Rui et al.: *A Survey on Low-Resource Neural Machine Translation*, in: Zhou, Zhi-Hua (ed.), Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21), s.l. 2021, 4636–4643; <https://www.ijcai.org/proceedings/2021/0629.pdf>.

Wolf, Thomas et al.: *Transformers: State-of-the-Art Natural Language Processing*, in: Liu, Qun/Schlangen, David (eds.), Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, s.l. 2020, 38–45; <https://aclanthology.org/2020.emnlp-demos.6.pdf>.

Yang, Yilin/Huang, Liang/Ma, Mingbo: *Breaking the Beam Search Curse: A Study of (Re-)Scoring Methods and Stopping Criteria for Neural Machine Translation*, in: Riloff, Ellen et al. (eds.), Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels 2018, 3054–3059; <https://aclanthology.org/D18-1342.pdf>.

Zhang, Biao/Haddow, Barry/Birch, Alexandra: *Prompting Large Language Model for Machine Translation: A Case Study*, in: Krause, Andreas et al. (eds.), Proceedings of the 40th International Conference on Machine Learning, Honolulu 2023, 41092–41110; <https://openreview.net/pdf?id=yWl0agiI0y>.

Zoph, Barret et al.: *Transfer Learning for Low-Resource Neural Machine Translation*, in: Su, Jian/Duh, Kevin/Carreras, Xavier (eds.), Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin 2016, 1568–1575; <https://aclanthology.org/D16-1163.pdf>, [08/04/2016].

Ressumé

Te chësta relaziun presentunse na modalité de traduziun automatica neuronala por le ladin dla Val Badia. La modalité neuronala se ghira n gran numer de traduziuns d'ejëmpl por che le sistem funzionëies indortöra. Dal momënt che la desponibilité de chisc dac paralei por le ladin é dër limitada vál debojëgn da abiné adöm chisc dac tolon sö tesć ma te un n lingaz, tuc fora en gran pert dal foliet edemal ladin "La Usc di Ladins". Nos adorun na implementaziun dl dizionar talian "Italiano–Ladin Val Badia" te *Apertium* por filtré y comedé i tesć y ince por cherié traduziuns temporanes. Tolon le *API* de *DeepL* laurunse spo fora chëstes traduziuns, ciaran dantadöt da les mioré dal punt d'odüda gramatical. Le *corpus* che vëgn insciö a s'al dé, vëgn adoré coche basa por deplü esperimënc. I insignun jö i modei *Transformer* bele dal mëteman inant y i adatun modei de traduziun che é bele, arjunjon resultac ezelënc cun trames les modalités. Nüsc esperimënc desmostra che chisc sistems neuronai funzionëia damí co i modei de traduziun automatica statistics y chi che se basëia sön regoles studiades cina dan da püch. I un metü a desposiziun i modei svilupá tres n'aplicaziun web. Implü unse cherié na plataforma por la revijiun costanta dl *corpus* por podëi mioré le model tres l'intervënt dla porsona tl *post-editing*.

Abstract

In this report we present a neural approach to machine translation for the Val Badia variant of Ladin. To achieve good results, neural models require a large number of exemplary translations on which they can be trained. The limited availability of such parallel data for Ladin makes it necessary to synthesise such data by using monolingual texts. We mainly use texts from the Ladin newspaper "La Usc di Ladins" as a basis for this so-called back translation. We translate these texts into Italian, using a rule-based system implemented in *Apertium*. Using *DeepL API*, we postprocess these translations and improve them, mainly at grammatical level. The resulting corpus serves as a basis for the different experiments we perform when we train models for this language pair. We train *Transformer* models from scratch and fine-tune pretrained models. With both methods we have achieved results that outperform the statistical and rule-based approaches to machine translation investigated so far. The models have been made available by means of a web application. Furthermore we have launched a platform for the continuous revision of the corpus to allow for the continuous improvement of the model through human post-editing.